博士論文 324

A Study on General Purpose Understanding System for Drawings

(図面画像理解の多目的化に関する研究)

指導教官 坂内 正夫 教授

東京大学大学院 工学系研究科 電子工学専攻

87126 魯 偉

1994年12月20日提出

Contents

Chapter 1

Introduction

1.1 Backgrounds

There are numerous types of drawings being used in all fields such as manufacturing, construction, designing and so on. Generally speaking, drawings are used for describing all types of information for objects in visually recognizable format. Dimension, shape, type, manufacturing procedure and so on are some examples of such information. All types of lines, symbols, and text are the main contents of drawings. Map drawing (city map, topographical map etc.), meteorological map, electric circuit drawing, mechanical drawing, logical circuit drawing, utility drawing (pipe layout, power line, gas distribution etc) are some examples of drawings. Up till now, tremendous number of drawings have been drawn and stored in many institutions/companies in the form of paper. Even when the CAD/CAM systems have been significantly improved in quality and functionality, and produced at lower cost, there are still paper-based drawings produced due to economical, presentational and technical limitations. There are also cases where drawings generated by one computer system need to be utilized in other system with different specifications and simple format conversion is difficult, so that utilization of the drawing has be conducted

by the paper based drawings.

On the other hand, when the number of drawings is small, their maintenance such as storing, sorting, searching and so on is relatively easy. But when the number keeps increasing, it takes much more efforts to maintain the existing drawings. Especially when the production cycle becomes shorter and shorter so that the information depicted in the corresponding drawings changes more and more frequently, the reuse, modification and updating of existing drawings also become more and more important. For paper-based drawings, this indicates the necessity of generating the whole drawing again. When all the drawings are in computer readable format, i.e. CAD data of polyline, surface and abstract attributes, the above operations can be carried out with ease. This also means that conversion of paper-based drawings to computer readable format is also becoming more and more in demand. There have been digitizing systems that make use of track balls or other similar devises for converting drawings into computerized data. But because of the quantity of information in drawings, this type of conversion usually requires long hours of labor work such as analyzing the contents of a drawing for inputting, tracing carefully the line segments, inputting the attribute information, linking the relational information and so on. The precision, efficiency and reliability of such kind of systems are far from being satisfactory[1], [2].

There have been many researches on recognition and understanding of paper based drawings and conversion of the drawings to computer readable format by making use of technologies such as image processing, pattern recognition, computer vision database and so on. Since drawings are usually drawn by following related regulations and constraints, they

are much less fuzzy than images of natural scenes and therefore possible to be processed at more abstract level [4]. In fact there have been some systems that can recognize/understand drawings of specific types and with good quality. On the other hand, drawings with bad quality and complex variation of specifications are still difficult to process by automated recognition/understanding system [3],[4]. As a result, manual operations with the assistance of human being are required. Generally, while images of drawings seem to be more intuitive and easy to understand than natural images, they are still difficult for computers to analyze. One of the main reasons is that the actual requirement for inputting drawings into computer, i.e. conversion of drawing image into computer readable format, is far complex than only identifying line segments, text area etc. In many cases, the information in a drawing has to be analyzed by integration of the depicted graphical elements and its corresponding regulation and constraints. Misinterpretation of any part due to either natural deterioration or human mistake in system configuration can lead to failure of correct conversion. Another reason is that drawings are generally designed for easy visualization of information with graphical representation. Although the result is supposed to be easily interpretable by human being, it is not necessarily as easy for computer to process from pattern recognition, vision and artificial intelligence point of view. Furthermore, drawings of different fields have different complexity, different drawing specifications, different storing method and so on. Therefore the quality, requirement for information retrieval and consequently processing strategy become different for different type of drawings. This research presents several approaches for more flexible and reliable construction of understanding system for drawings, so that the existing systems can be more easily adopted

for different type of drawings. The first approach is to separate knowledge for recognition and understanding from the inputting system and describe them with production rules which drive a blackboard based inference engine for the desired recognition/understanding process. The second approach is proposed for improving the efficiency, reliability and flexibility of manual operations, an interactive recognition system is proposed which makes use of processed objects for generation of recognition proposals. The third approach for improvement of flexibility is to assist the acquisition of recognition rules by applying machine learning technologies, with necessary improvement of the existing algorithms. In order to verify the effectiveness of the proposed methods, experiments have been conducted to apply the proposed approaches to typical drawings.

1.2 Configuration of the Thesis

The configuration of this thesis is as follows:

Chapter 1 discusses the objectives and configuration of this thesis.

Chapter 2 discusses the requirement, problems and previous works on recognition and understanding of drawings.

Chapter 3 discusses a flexible way of system construction based on production rules and blackboard model.

Chapter 4 discusses an efficient and reliable interactive recognition system for processing graphical elements that are rejected by automated recognition process.

Chapter 5 discusses a recognition rule acquisition system as a supplement of other technologies for easy construction and maintenance of drawing recognition/understanding sys-

tem.

Finally Chapter 6 summarizes the contents of this thesis.

Chapter 2

Previous Works

2.1 Understanding of Drawings

Objective

As mentioned in Chapter 1, with the fast progress of CAD/CAM/FA systems, the demand for the digitization of related resources is also increasing. Among the related resources, existing paper-based drawings of all types are playing very important roles and therefore need to be digitized for efficient applications in computerized environments. To meet this demand, many researches on digitizing system for drawings have been conducted by adoption and improvement of technologies such as image processing, pattern recognition, artificial intelligence and data base [1], [2], [3],[4],[5].

The purpose of Recognition/Understanding of drawings is the retrieval of geometrical information such as shape, dimension, structure and so on from the graphical elements such as all types of lines, text, symbol and so on. The process is preferably performed automatically without any efforts by human. But the quality of the existing drawings varies with many factors so that not all can be processed automatically. The information obtained by recognition/understanding process will then be converted to the format of

other applications, typically all types of database system. Generally speaking, the actual information to be retrieved can be summarized as follows:

1. all types of symbols: in city map drawings, symbols indicating types of geographical objects such as building, hospital, school, rail road, field, forest etc.; in mechanical drawings, dimension related symbols such as arrow, text string, symbol of manufacturing procedure and so on.

2. text: indicating name, dimension of the graphical elements in a drawing

3. all types of lines: line width, types of line such as solid line, dotted line, chain line, hatching line etc.

4. structure: shape of all sub-parts of an object, dimensions, relationship between sub-parts etc.

Here, item 1 and 2 require pattern recognition techniques to *recognize* them, item 3 and 4 require comprehensive knowledge from geometrical information to constraints or regulations of drawings in order to *understand* them. For drawings such as design drawings, the 100% correct retrieval of shape and dimension information is required. Whereas for drawings such as circuit drawing, only the type of parts, the connection between them are required.

Drawing Information

Information contained in drawings can be divided into two groups: explicit one expressed by the graphical elements illustrated above and implicit one derived from the regulations and constraints of related standards. The detailed discussion of drawing information will be

easy				*difficult*
great	←	distribution of drawings	→	small
good	←	quality of drawings	→	bad
clear	←	regulation of drawings	→	fuzzy
weak	←	requirement for fairness	→	strong
weak	←	hierarchy of information	→	strong
strong	←	regularity of object	→	weak

Table 2.1: Difficulty of Understanding Drawings

addressed in Chapter 3. Here, the characteristics of drawing information can be summarized as follows:

1. variety of drawing type
2. specifications vary with regulations and even authors
3. quality of drawings varys with the type of paper, drawing tools and maintenance method
4. use of drawings differs with purpose of drawings

Due to the complexity of drawings, it is important to have efficient descriptions of drawing information and corresponding inference engine that makes use of the descriptions to recognize/understanding drawings.

Table2.1 shows the relationship of difficulty of understanding drawings with the characteristics of them [2]. For drawings that are relatively easy to understand, the variety of type and specification can be covered by higher modularity of processing systems. For drawings that are relatively difficult to understand, fully automated recognition/understanding becomes difficult, so that manual operations become indispensable. Therefore, improvement

of manual operations is also one of the important fields to be studied.

2.2 Preprocessing

Bitmap to Vector Conversion

The first step toward recognizing/understanding a paper based drawing is to convert it into digital imagery by an image scanner. The scanner can be a color one when the drawing is drawn in more than one color, or a monochrome one when the drawing is drawn in only one color, typically black on white paper. The result of scanning is a bitmap digital imagery data consisting of a 2 dimensional arry of pixels which have either light intensities or colors. Since the members in such arry only represent local information, a small region in the original drawing, with the area depending on the scanning resolution, more abstract information must be retrieved before more complex analysis of the drawing can be conducted. For imagery of natural scenes, such abstract information can be texture, shape, color or intensity and so on. For drawings, the most fundamental information is line segments and in some cases filled area.

To retrieve line segments from a drawing, the bitmap imagery drawing is first converted to binary imagery, with the pixel belonging to line segments converted to 1, others 0. Then the skeleton of the obtained binary data is computed to get the geographical representation of the line segments, i.e. start point (x_s, y_s) and end point (x_e, y_e), instead of a braid of pixels with the value of 1. There are mainly three approaches to obtaining the skeleton of line segments:

1. thining based on run-length code

2. thining based on iterative template matching

3. contour tracing and core vector formation

The first approach makes use of run-length code to obtain the skeleton of lines. The skeleton is taken as the middle point of the run length. By this algorithm, the image can be processed by one scan only. But broken line will occur if the image is scanned in one direction. This can be solved by combining the results scanned at both horizontal and vertical directions. Yet for line drawings of complex shape, the combination becomes difficult.

The second approach aims at obtaining the center of black pixel blocks by iteratively applying predefined templates to the binary image data. Each template is usually a 3×3 window which trys to identify the contour pixel of a line and if matched, delete the pixel in the center from the drawing. The main problem with this approach is the distortion at intersection of lines and long computational time. The time consuming problem becomes worse when extra efforts have to be made for actions such as reduction of the distortion at intersections, noise handling and so on. There are algorithm for speeding up the thinning algorithm by increasing the size of template[45]. But this type of improvement usually is at the cost of fairness of result. The best solution to solve the problem of computation time is its implementation by hardware[44]. Another common problem with both approach 1 and 2 is that the filled area will be corrupted after thinning. For this problem, application dependent approaches have to be taken to compensate the lost information. For example, thinning algorithm have been specifically proposed for recognition of Arabic text[46]. This is because the interpretations of dots, "T" joint and so on are significantly different from

other applications and have to be preserved well.

The third approach first traces the contour line of graphical elements from lines to all types of symbols. The obtained contour chain code is then approximated as contour polylines called *contour vector* by polygonalization algorithm to be discussed below. Finally, the median lines are drawn from contour vector pairs that are within the range of line width and have approximately opposite directions. The advantages of this method are (1) high speed implementation even purely with software based systems, since the vectors are processed once and the searching of pair vectors are realized with an efficient spatial data structure[13],[14],[64] (2) the filled areas can be preserved by their corresponding contour vectors[63]. This approach is adopted in this research and the details will be discussed in Chapter 3.

Feature Retrieval

After vectorization, the line segments in the original bitmap images now become closer to their geometrical meaning. But they are still not good enough for recognition or understanding of drawings, since most of the drawings consist of more abstract graphical elements rather than line segments only. In order to make the retrieval of the graphical elements easier, the line segments have to be classified into more abstract objects by their geometrical characteristics. Three of the main techniques are polygonalization of vectorization result, retrieval of prominent point and arc fitting.

1. Polygonization

   The most fundamental information in a drawing is line segments. Polygonization is to convert the thinned line segment or traced contour line segment, i.e. a list of

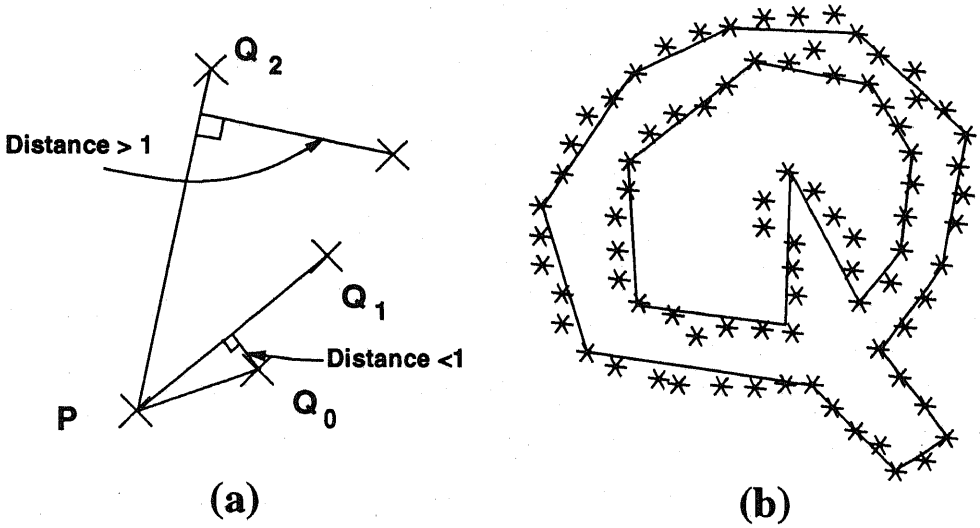


Figure 2.1: (a)Approximating a line using Euclidean distance and (b) result of applying (a) to the contour line of letter Q

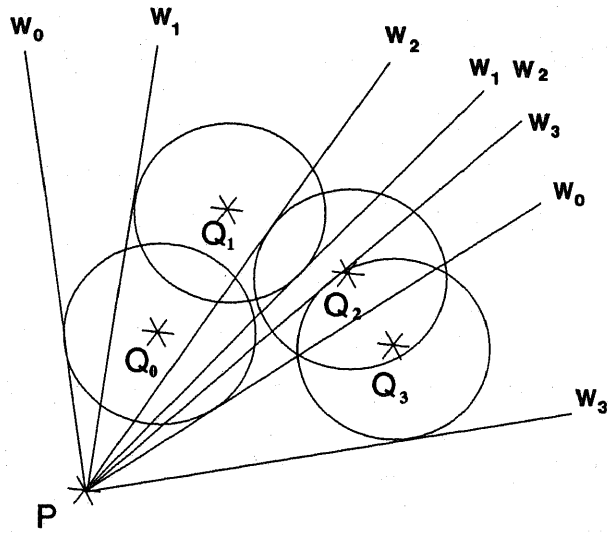


Figure 2.2: Polygonization by tangential wedges

connected pixel coordinates, into geometrical line segments i.e. a polyline or polygon which optimally approximates the original pixel list. It has three functionalities: (1) reduction of data, usually by an average of 15:1, (2) delimitation of perturbations in the edge, and (3) easier recognition of graphical elements.

Fig.2.1 shows one typical approach proposed by Jimenez[52]. The proposed thinning process starts from a pixel P and repeats the following procedures when the pixel list that is connected to P is $[Q_0, ..]$ and $i = 0$:

step 1 draw a line from Q_{i-1} to line PQ_i

step 2 calculate the distance D from P to Q_{i-1}

step 3 iff $D < 1$, then $i \rightarrow i + 1$, goto step 1

step 4 Output Q_i as vertex of the approximated polyline, $P \rightarrow Q_i$, repeat step1 until the pixel list is empty

When the distances obtained during the above procedures can be assumed to be non-decreasing function of the number of pixels between P and Q, then a binary search can be used to find the Q.

Another approach to polygonization proposed by Slamsky is shown in Fig.2.2, where a wedge that is tangents to circles around each point is used to determine the point Q[53]. It is a similar version of Williams' approach[12]. The wedge is modified with each new point such that it forms the intersection of the wedges from all points so far. The searching stops when the intersection of wedge doest not exist. For example, the intersection of tangential wedges of Point Q_0 to Q_2 is W_2. When Q_3 is under

consideration, the corresponding wedge does not intersect with W_2. Therefore the Q_2 becomes the vertex of the polyline and the polygonization starts from Q_2 again.

Since the result of polygonization is an approximation of the original contour line or skeleton line, the geometrical feature of the original data can be lost. Therefore, other algorithms that aims at preserving the original geometrical features are also proposed. For example, the algorithm proposed by Aoyama and Kawagoe makes use of circle fitting to recover smooth connecting points of arc and line[54].

2. Retrieval of dominant points

When recognizing the shape of a graphical element by its sub-parts, it is necessary to identify the most dominant points and use them to divide the element into sub-parts. A dominant point of a curve is the point where the curvature changes comparatively dramatically. They are usually the candidate points where a curve is divided into sub-curves. Polygonization stated above is one of the most commonly used approach in detecting dominant points. But the result by polygonization sometimes is not stable, usually reflected at sensitive distortions to local noises. Besides, when the curvature of a curve changes in a wide range, a vertex of the polygonized result is not necessarily a dominant point. One recent improved approach is to detect dominant points by curvature guided polygonal approximation [55],[56],[58].

When a planar curve is represented by a set of points in parametric form,

$$(x(t), y(t)) \in \mathbf{R}^2,$$

where t is the path length along the curve. The curve is first smoothed with Gaus-

sian filter, which is the same as convolving $x(t)$ and $y(t)$, respectively, with one-dimensional Gaussian filter,

$$h(t,\omega) = \frac{1}{\sqrt{2\pi}\omega} exp\{-0.5(t/\omega)^2\},$$

where ω is the width (spatial support) of the filter. Then the smoothed curve can be expressed as $(X(t,\omega), Y(t,\omega))$, where,

$$X(t,\omega) = x(t) * h(t,\omega),$$

$$Y(t,\omega) = Y(t) * h(t,\omega),$$

and * indicates the convolution operator. The curvature of the smoothed curve can be derived as,

$$K(t,\omega) = \frac{\dot{X}\ddot{Y} - \dot{Y}\ddot{X}}{(\dot{X}^2 + \dot{Y}^2)^{3/2}}$$

The points with local extreme (positive maximum and negative minimum) value of the above curvature are taken as the initial dominant points. The initial points are then used by split-merge method to find the full nominal points. Fig.2.3 illustrate a result of dominant point detection from the contour line of wrench. By this approach, the dominant points obtained are less sensitive to orientation and scaling variation.

2.3 Graphical Element Retrieval

After retrieving the fundamental graphical features of the vectorized lines, the shape analysis can be conducted to retrieve the graphical elements consisting of line segments. Usually

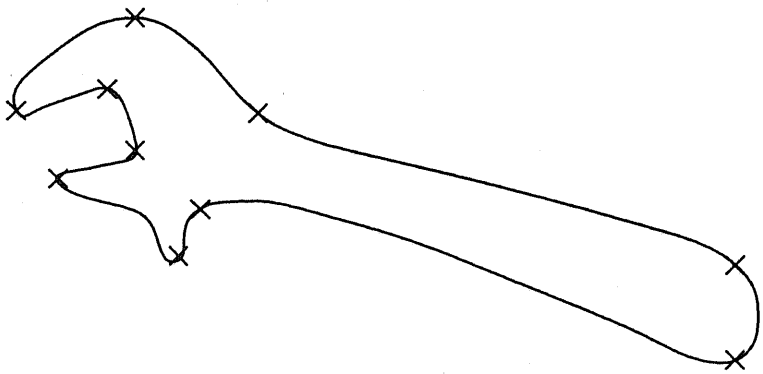


Figure 2.3: Dominant point detection by curvature guided polygonal approximation

the purpose of shape analysis is to classify the target graphical elements into several classes according to their geometrical attributes, which are statistically distinguishable since different class has one or more different geometrical attributes.

When a graphical element is a polygon, it can be analyzed by the attributes of the polygon. The most commonly used attributes of a polygon are area, peripheral length, gravity center, area of convex polygon, peripheral length of convex polygon, projection to X-axis/Y-axis, circumscribing rectangle both in X-axis direction and elongation axis etc.[69]. These attributes are effective for classification of graphical elements under certain conditions such as constant scale, constant orientation etc., depending on the design objectives.

When the shape of graphical elements belonging to different classes are difficult to distinguish by the above mentioned attributes, more robust attributes become necessary. Generally, more robust attributes are derived from the dominant points discussed in Section 2.2. Some examples will be discussed below. To save space, the equations have been reduced to the minimum. More details can be found in the corresponding references.

Milios applys curvature processes and dynamic programming to sub-parts of contour line of graphical elements for their shape matching[17]. The sub-parts are obtained by dividing the contour line at the dominant points obtained with the approaches discussed in Section 2.2. When two graphical elements x and y are represented by their cyclic sub-part lists such that $x = a_1a_2\cdots a_n, y = b_1b_2\cdots b_m$. In order to apply dynamic programming algorithm, the cyclic sub-part list must be turned into open ones. This can be done by finding the best matching pair of segments and making these segments the starting segments of the open sequences derived from the cyclic ones. The cost array $D(i,j)$ can be defined as,

$$D(i,j) = \begin{bmatrix} D(i-1,j-1) + \gamma(a_i, b_j) \\ D(i-1,j) + \gamma(a_i, nil) \\ D(i,j-1) + \gamma(a_i, nil) \end{bmatrix}$$

where,

$$\gamma(a_i, b_j) = d_1 d_2,$$

$$\frac{1}{d_1} = \frac{\int_{\infty}^{-\infty} \kappa_{a_i}(\theta)\kappa_{b_j}(\theta)d\theta}{\sqrt{\int_{\infty}^{-\infty} \kappa_{a_i}^2(\theta)d\theta \int_{\infty}^{-\infty} \kappa_{b_i}^2(\theta)d\theta}},$$

$$\frac{1}{d_2} = 1 - \frac{|a_i| - |b_j|}{|a_i| + |b_j|},$$

and $|a_i|, |b_j|$ denote the lengths of the corresponding segments, $\kappa_{a_i}, \kappa_{b_j}$ the curvature of extended circular image, $(i,j)th$ element of the array is equal to the optimal cost of matching $x(i) = a_1a_2\cdots a_i$, the first i segments of x with $y(i) = b_1b_2\cdots b_j$, the first j elements of y. Searching through the cost array for an potimal path from $D(0,0)$ to $D(n,m)$ will then yields the segment associations between two graphical elements.

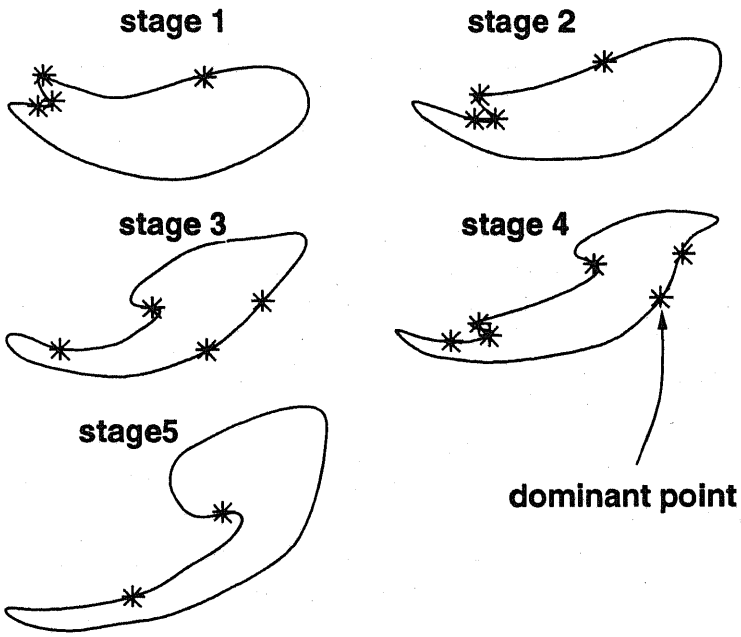


Figure 2.4: Silhouettes of cloud at different stage of cyclogenesis evolution

This approach has been successfully applied to matching of cloud silhouettes for weather forecasting. In this application, the graphical element to be analyzed are the corresponding part of cloud silhouettes at different stages of evolution during cyclogenesis. Fig.2.4 shows some examples of silhouettes of actual cloud. The cloud changes from leaf-shaped one at stage 1 to comma-shaped one at stage five. The "*" mark indicates the dominant points. Even though the number of segments in every silhouette is sometimes different, the clouds are successfully matched by the dynamic programming algorithm. The result can be further matched with the descriptions against knowledge bases of various meteorological phenomena.

Ansari takes another approach of using dominant points for shape matching[56]. In his research, a dominant point is called *landmark*. A set of three neighboring dominant points

are used to form a triangle. The sphericity of a triangular transformation which maps a triangle to another triangle is used as a measure of the similarity between the two triangles. Here, the sphericity of a diffeomorphism [1], $g : \Omega \leftarrow \overline{\Omega}, (\Omega, \overline{\Omega} \subset R^n)$, for $x \in \Omega$, is defined as

$$Y_g(x) = \frac{(det\,(g''(x)g'(x)))^{1/n}}{\left(\frac{1}{n}tr(g''(x)g'(x))\right)}$$

where $x = [x, y]^t$, $det()$ is the determinant of a matrix and $tr()$ the trace of a matrix respectively. It has been shown that the diffeomorphism is affine transformation, and the sphericity defined above is rotation, translation and scale invariant.

When the classes of the graphical elements are known and their contour lines are available, the dominant points of contour lines of each class can be detected and stored as a model base. When an unknow element is presented to the recognition system, its dominant points are first detected. The hypothesis of a model object in a scene is then made by matching the model dominant points with the scene dominant points. A table of compatibility is constructed between the sequence of model dominant points and the sequence of scene dominant points. The row index i corresponds to a model dominant point while the column index j corresponds to a scene dominant point. The (i, j) entry of the table is the sphericity value of the triangular transformation mapping the model dominant points and its two adjacent dominant points to the jth scene dominant point and its two adjacent dominant points. To avoid brute-force searching for correspondence between model and scene, dynamic programming procedure is formulated to achieve the matching as Milios does.

[1] A *diffeomorphism* is defined as a continuous one-to-one mapping such that the inverse mapping is also continuous and both the mapping and its inverse have continuous partial derivatives.

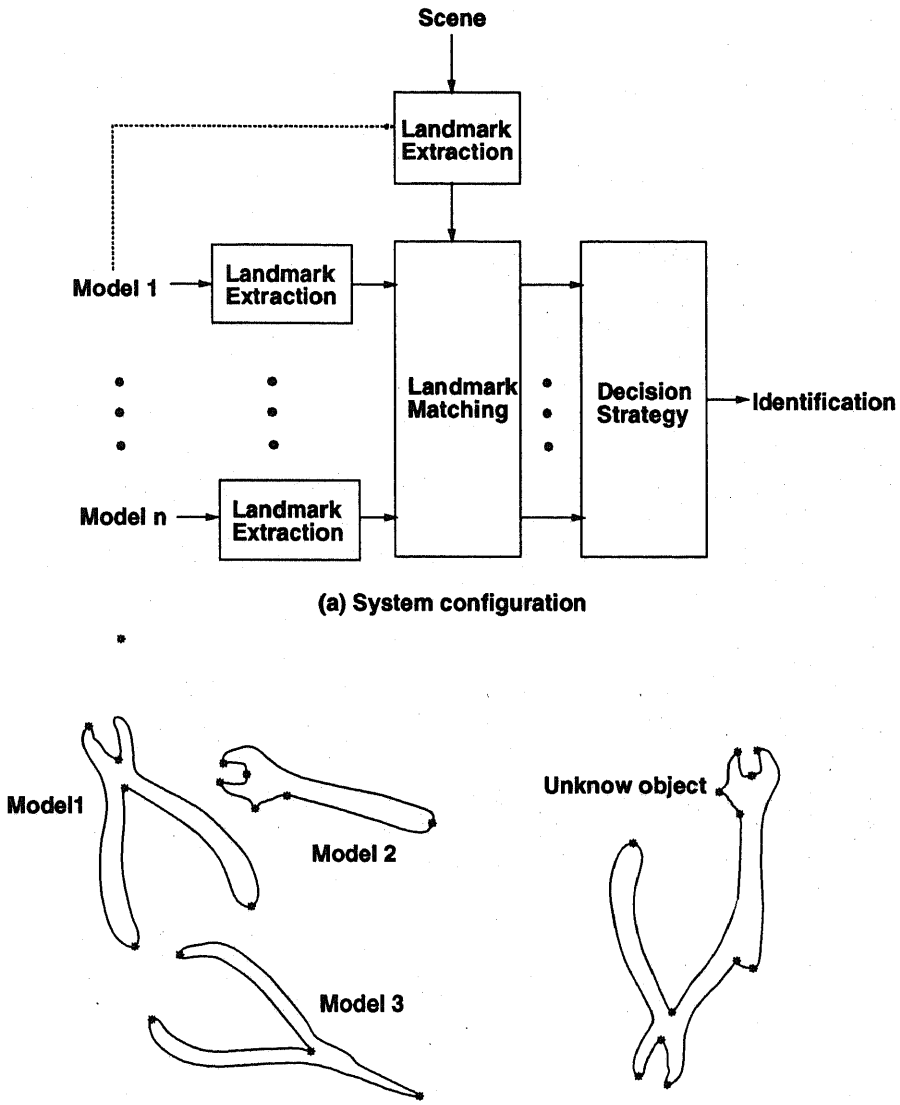


Figure 2.5: Partial shape recognition by dominant points and their sphericities

The configuration of the recognition system is shown in Fig.2.5(a), and some of the model objects and one actual object to be recognized are shown in Fig.2.5(b). The "*" marks are dominant points detected by an algorithm similar to that stated in Section 2.2. The recognition target consists of two overlapped independent objects. Since Ansari's approach is based on trio-dominant-points, as long as more than three dominant points are present, it is possible to have a plausible match with the model. For the objects such as those shown in Fig.2.5, the objects can be correctly recognized even part of the object is occluded. From the illustrated examples, it is obvious that the algorithm is independent of rotation, translation and scale.

It should be noted that both Milios and Ansari's approaches require that the contour line of the target object should be available and the objects differs only by the shape of their boundaries.

Fahn and Wang make use of topological information for retrieving symbols of electronic components in circuit diagrams[19]. The diagrams are processed mainly in five steps in order to retrieve the component symbols:

1. vectorize the drawing into line segments
2. detect the dominant points
3. detect primitive sub-parts of graphical elements. There are six classes: resistor, inductor, straight, doglegged, zigzag and curved
4. generate compound segments
5. combine compound segments

Here, step 3 makes use of the following criteria:

Criterion 1 The existence of long approximated picture segment which contains a series of vectors $V_1 \cdots V_k$ and $k \geq 4$ such that $V_i \in \{V_a \cdots\}$ and $T_{HS} < L(V_i) \leq T_{VS}$ for $i = 1, 2, \cdots, k$.

Criterion 2 The existence of a series of trends $T_1 \cdots T_{k-1}$ which is derived from every couple of consecutive vectors in $V_1 \cdots V_k$ such that $T_i = \pm 2, \pm 3$, or ± 4 for $i = 1, 2, \cdots, k-1$, and $T_i \cdot T_{i+1} < 0$ for $i = 1, 2, \cdot, k-2$

Criterion 3 The number of vectors $V_a, \cdots, V_c$

Criterion 4 The existence of a series of trends $T_1 \cdots T_m$ which are derived from every couple of ordinal odd or even vectors in $V_a \cdots V_c$ such that $T_i = 0$ or ± 1 for $j \in \{1, \cdots, m\}$

And some of the topological information used in step 4 are as follows:

(a) Loop relation

(b) Symmetrical relation

(c) Smooth relation

(d) Uneven relation

Another representative approach is to make use of statistical attributes and hierarchical neural networks for shape analysis. Cheng and Khan proposed to use invariant moment extraction for symbol recognition[25]. While this approach requires perfect separation of

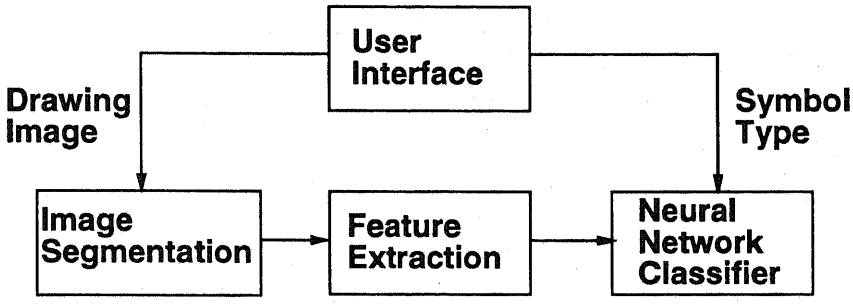
the candidate of target symbols, it realizes recognition of symbols in electronic circuit drawings with high recognition rate. The invariant moments used in their approach are

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - (3\eta_{21} + \eta_{03})^2] \\
&\quad +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{03} + \eta_{21})
\end{aligned}$$

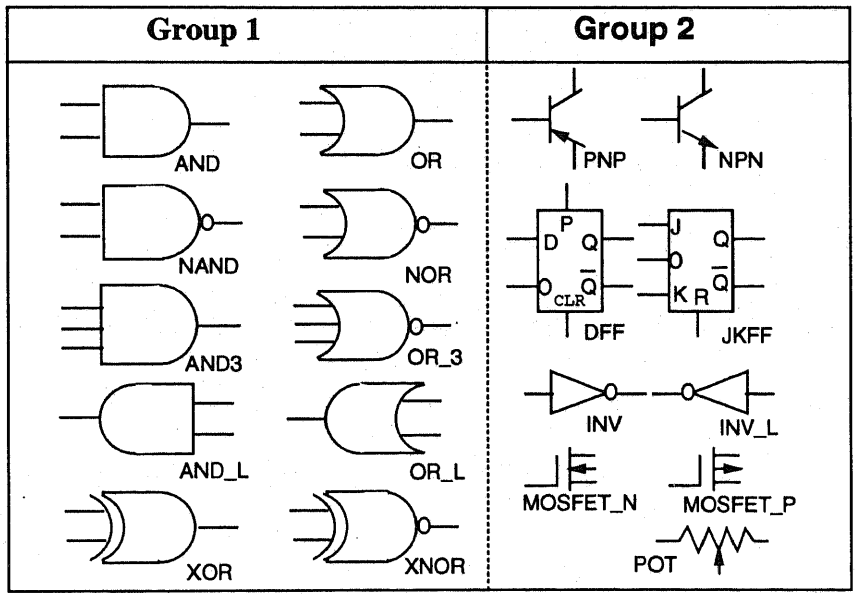
Where all the moments are normalized by the central moment for size invariant as follows:

$$\eta_{pq} = \mu_{pq}/\mu_{00}^{\gamma}, \quad \gamma = (p+q)/2 + 1$$

The basic system configuration and part of the symbols are shown in Fig.2.6(a) and (b). As teacher data, the sample symbols are first preprocessed by three basic functions, *autocropping, centering and normalization.* Then the invariant moments stated above are extracted. A two layer neural network is used for training. The first layer groups the symbols into 4 sub-groups. The second layer realize the final recognition. The recognition rate to teacher data is 98.5% and text data 89%. There are also approaches based on contour pixel list or even original bitmap image. The approach reported in reference [16] makes use of hidden Markov model to obtain rotation and translation invariant representation for classification of shapes with contour perturbation. Such approach is effective to elements



(a) System configuration



(b) Examples of symbols

Figure 2.6: Recognition system for symbols in electronic circuit drawings

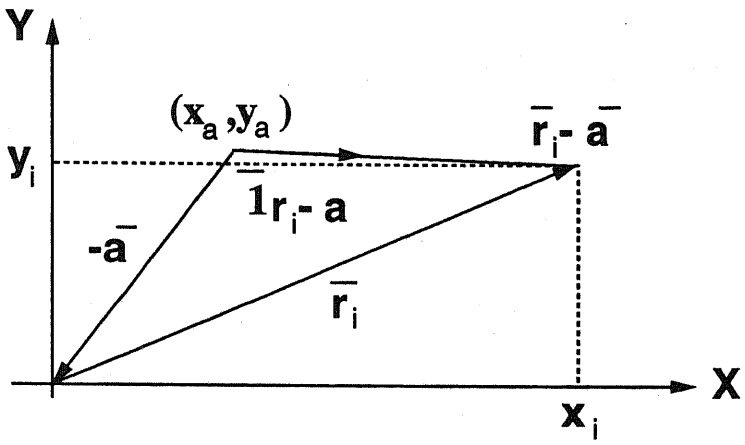


Figure 2.7: A data point(x_i, y_i), the current arc center and the radius R

with contour perturbation and a moderate amount of occlusion. The approach reported in reference [57] makes use of 3×3 directional operation templates and mathematical operations using the result of template to realize a parallel symbol recognition system. The experimental result shows that the parallel template approach can recognize all kinds of symbols under moderately noisy situations. Since the approach is parallelism, it is possible to realize efficient hardware implementation. The problem with this approach is the difficulty of identifying the suitable operations for recognizing desired pattern, because the feature obtained from the template can only reflect local information and is not intuitive.

The technique of arc fitting is also very important in retrieval of graphical elements. Most of drawings use either a complete circle or an arc of different radii or starting/end point for symbolic description of some objects. In the case of drawings for shape description, such as mechanical drawing, circle or arc is usually a very important type of object shape. Therefore, the algorithms for detecting circle/arc has long been studied. Landau proposed a general approach to estimation of a circular arc center and its radius[21]. When a set

of points $(x1, y1), \cdots, (x_N, y_N)$ is given, the radius vector $\overline{r_1}, \cdots, \overline{r_N}$ to each point and a tentative center $\overline{a} = (x_a, y_a)$ can be defined as shown in Fig.2.7. The optimum radius of the estimated circle/arc and the center can then be calculated by

$$R = \frac{1}{N} \sum_i |\overline{r_i} - a|,$$

$$\overline{a} = \frac{1}{N} \sum_i (\overline{r_i} - R\overline{1}_{r_i - a})$$

which are obtained by minizing the error of estimated radius and center instead of LMS error between the given set of data points and the estimated circle. The actual center and radius can be calculated by the following iteration:

(a) Initiate $\overline{a}_k (k = 0)$

(b) Substitute $\overline{a}_k$ in the above equation and find R_k

(c) Substitute $\overline{a}_k$ and R_k in the above equation and find $\overline{a}_{k+1}$

(d) If $|\overline{a}_k - \overline{a}_{k+1}|$ is less than some positive, then exit, else, goto step (b)

This algorithm can converge to the same result, independent of the initial center.

Rosin proposed an algorithm for arc detection using polygonization result[22]. When the possible maximum and minimum radii are known, his algorithm can detect arcs without any specific error threshold. After a set of point list is isolated for arc detection, the algorithm finds out the optimum radius and arc in the following 6 stages:

(1) transform the coordinates of the vertices so that the center of the line joining the two outer points lines at the origin and the two outer points lie on the x-axis.

(2) compute the error with the center of hypothesize circle center at the origin (0,0) and either side of the x-axis (0,1) and (0,-1) to decide the direction of the hypothesized arc.

(3) compute the mean y coordinate of all the vertices to judge the angle subtended by

the arc, i.e. whether the angle is greater or equal to 180 degrees or smaller.

(4) set up the search space in which to search for the minimum error. This can be decided by domain knowledge or the resolution of the image.

(5) search for the minimum error by recursive binary splitting algorithm.

(6) compute the parameter of the arc, i.e. center, radius and length.

2.4 Structural Information Retrieval

After the line segments are vectorized, their related fundamental feature calculated and the graphical elements retrieved, the further *understanding* of the drawing can be conducted at more abstract level based on the processing results discussed in Section 2.2 and Section 2.3. There have been many researches aiming at understanding of all types of drawings by application dependent approaches. There are a few bitmap image based approaches such as the one Agui and Maesaki proposed [10]. Their approach makes use of a template moving along the graphical object and matching the desired patterns. While this type of approach can be noise and distortion insensitive to certain extent, it is no doubt time consuming. The template designed for the understanding process is also complex and not intuitive. Therefore, most of other approaches are based on the result of preprocessing process discussed in previous sections.

When a drawing does not have intricate structures, the main task for understanding usually only requires tracing of branches, detecting linkage between broken elements and so on, as in the case of inputting system for drainage networks drawings[23]. Drainage drawing is for indicating the distribution of drainage routes, and usually obtained by viewing the

soft copy display and manually tracing main, low lying collector channels with shorter feeder channels coming in from higher elevations. In the implemented automatic tracing system, the candidate of drainage points are first retrieved from stereo image. Then the extraneous points are clustered out by checking the vicinity to other points, the total number of connected points. The clustered candidates of drainage point are converted into vector form by tracing along the chain of the candidates and linking supposedly broken chains. Finally, the hierarchy structure of the drainage network is derived from the tracing and linking result in a tree format. Therefore, the task of retrieving drainage network is a process of structured vectorization. The main criterion used for tracing and linking are based on the resolution of the image and the possible minimum length of drainage.

When the task is for fair display or printing, the main tasks are recognition of lines and efficient storing of the lines such as the research conducted by Ablameyko[8]. In their research, the line segments are first vectorized as line vectors. Then a labeling process labels the line segments according to the spatial relationships such as joining, crossing, gap, lying inside, lying in neighborhood and isolated. In this way, the line information can be represented in a compressed and structural manner so that when necessary, other application can easily retrieve information of higher level.

The rest part of this section will discuss other existing approaches for more sophisticated requirements.

Syntactical Approach

When the drawings contains more than just line segments and the meaning of lines is not unique, more sophisticated processes become necessary. One fundamental approach

emphasizes on describing information depicted in drawings in syntactical fashion, and then applys the syntactical algorithms of pattern recognition technology. Generally, this type of approach trys to use the 5-tuple: $T = (N, \Sigma, \Delta, R, S)$ to describe the relationship of the graphical elements, where

N is a finite set of nonterminal symbols

Σ is a finite input alphabet

Δ is a finite output alphabet

R is a finite set of rules of the form $A \rightarrow \alpha, \beta$, where $A \in N, \alpha \in (N \bigcup \Sigma), \beta \in (N \bigcup \Delta)$

S is a special non-terminal in N, namely the starting symbol.

The approach reported in reference [59] automates part of the recognition tasks of digitizing map drawings by using the above grammar for describing numbers with one decimal, correcting some type of classification errors in character recognition and finding legal descriptions of parcel codes on cadastral maps. The main reason that application of syntactical approach to the three tasks is possible is mainly because of the fact that fixed and simple syntax exist for decimal number, text and parcel codes. For example, the characters "B", "l", and "2" might be wrongly classified as "8", "1" and "z" respectively. At the same time, according to the context of these characters, there can be some contradictions when the classification result is wrong. Converting these contradictions into production rules in the grammar can then realize the correction of the corresponding errors. For other tasks such as recognition retrieval of line segments, classification of lines and recognition of characters are still difficult for syntax analysis.

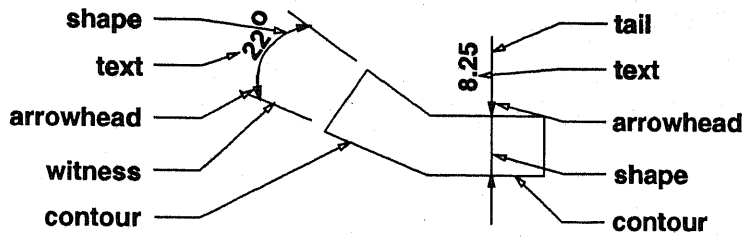
Dori makes use of syntax analysis for recognition of dimension information in mechanical

drawings[60] . Generally, mechanical drawings are drawn by strictly following the standards such as ISO in Europe, ANSI in America and JIS in Japan. A *dimension-set* in mechanical drawings is a set of graphic entities for description of measure between two turning points on the contour of the object's projection. Following the standard of ISO, the graphical components used for describing dimension-set is divided into contour, witness, shape, tail, arrowhead and text as shown in Fig.2.8(a). Six *Associators*: bind, tip, point, pierce, and guide are defined for describing the relationship between the components(Fig.2.8(b)). A dimension grammar is then defined by web structures with the components and the associators. An example of production is shown in Fig.2.9. When the components are correctly retrieved in the graphical element recognition process, the meanings of the corresponding dimension-sets can be analyzed by the the pre-defined web grammar. All the web grammars defined in [60] are initiated by arrow head. The problem with this approach lies in the construction of the grammar, which will always fail whenever the retrieval of arrow head fails. In this case, more complex grammar will become necessary. While web grammar can theoretically describe almost any situations, the transparency and intuitiveness are difficult to retain.

Interactive Processing Systems

No doubt that the ultimate goal of recognition/understanding system for drawings is to automatically convert all the information depicted in the drawing to computer readable data. Whereas as far as the technologies are going, while the ability of the implemented system is being improved significantly, realization of fully automated system is still an extremely difficult task. The practical inputting systems for drawing still require the as-

name	symbol	function in dimension-set
contour	C	sites between which the dimension is measured
witness	W	guides to the contour component
shape	S	represents the geometric shape of the dimension
tail	L	extension of shape when shape is small
arrowhead	A	points at a contour or at witness
text	T	specifies dimension type (e.g. angular) and value



(a) Grouping of compoents of dimension-sets

name	symbol	function in dimension-set
bind		binds text to shape or tail
tip		arrowhead overlaps shape or tail
point		arrowhead points at shape
pierce		witness or shape are pierced by arrowhead
guide		witness guides to contour

(b) Grouping of relationship between components

Figure 2.8: Fundamental definition for understanding of mechanical drawings

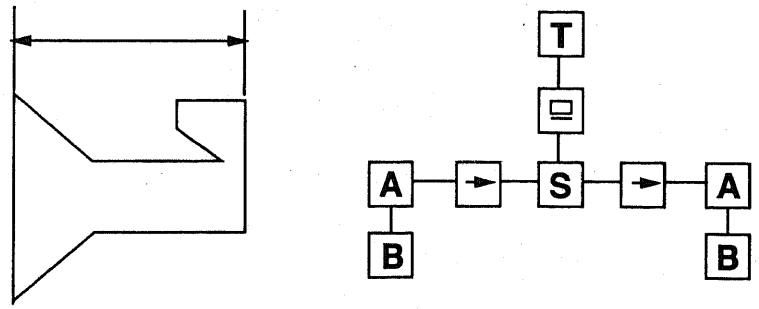


Figure 2.9: An example of production for a symmetrical dimension-set

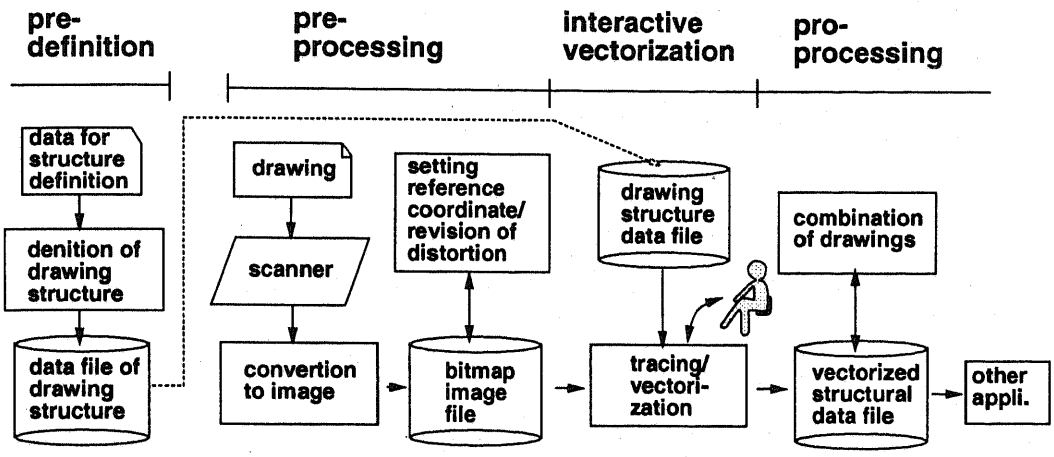


Figure 2.10: The configuration of interactive inputing system

sistance of human being. With the rapid improvement of computer systems' performance, the recognition and understanding process can be realized in shorter and shorter time. But when there exist graphical elements that can not be processed automatically by computer systems, they have to be processed by the assistance of human being, in interactive manner. By statistics from the existing systems, the time spent on interactive processing by human being is much more than required by automated process. For some type of drawings, the graphical objects are hand drawn, the density can be over dense and the overall structure irregular, so that the best way to digitize them is by manually tracing the objects or input their attributes. The time for manually digitizing a drawing can take dozens of hours. When considering the number of existing drawings, which is usually in hundreds or even thousands, the improvement of efficiency of interactive process is a non-trivial problem.

Yamagawa proposed an interactive inputting system for map drawings in which operator gives instructions of vectorization by interactive process[20]. Fig.2.10 shows the configuration of the interactive inputting system. The "interactive vectorization" process realizes

conversion of bitmap image data to vector data. The graphical elements to be vectorized and the corresponding attributes are specified by operator. The contour line of the bitmap data is first traced and polygonized before entering interactive process. When the operator specifys a graphical element to be vectorized, the system performs a spatial search to find the nearest contour vector from the specified point. In this way, the specification point need not lie precisely on the line image and the operation can be done by only viewing the same screen, which reduces the effort of locating the precise digitization points. This is extremely important since when using a manual digitizer, the operator has to view the paper drawing and the mesh displayed on the screen separately to ensure precise location. At the same time, the system can also vectorize distorted or overlapped lines especially small arcs or short polylines that are difficult for automated process. The verification of vectorization result is also easy and reliable since the result is displayed over the original bitmap image. Nagano and Kanechika proposed a similar interactive map inputting system by manually giving instructions for error correction or hints for recognitions[47]. While these types of approaches can improve the efficiency and reliability of drawing inputting, the human efforts required by the system is still overwhelming and there is also redundancy in repetitive operations for similar situations.

Maehara and Shibayama proposed another interactive system for inputting facility maintenance drawings[48]. The target drawings for inputting are not drawn precisely and have numerous complex overlappings between graphical elements, with the quality having deteriorated during long term storage and distortion incurred by multi-generation photocopying. Therefore, it is not plausible to employ automatic process for inputting this type of draw-

ings. To realize efficient and reliable digitization, they adopted a sketch-guided approach which requires the operator to draw rough sketch of the target drawing and let the system do the precise recognition of corresponding graphical elements according to the sketch and rearrange the sketch according to the recognition result. This process might need some trial and error when the automatic rearrangement of the sketch is incorrect. All the information depicted by text strings are inputted manually. The connection between the text strings and the related graphical elements are also specified by operator. After inputting all the graphical elements and related text information, the system will perform automatic checking of contradiction and error in the result and generation of data for other applications. By this approach, the time required for inputting a drawing has been reduced to 23% of that required by pure manual inputting. The problem with this approach is that the same kind of trial and error process has to be repeated even if similar situation has been processed.

Rule Acquisition

In order to perform recognition/understanding of drawings, all the approaches discussed above require some type of rules that are usually manually generated by observing the characteristics of the target object, selecting the most discriminative attributes and finally the distribution range of the selected attributes. The process of rule generation requires many trial and errors and becomes difficult when the number of classes and attributes increase. There have been researches on efficient acquisition with the aid of computer.

Yachida proposed a computer vision system that has a user interface module for easy updating of recognition rules for images of mechanical parts[33]. In his system, the rules for

recognizing target images are represented by a set of attribute list and their corresponding distribution ranges. The attributes are constructed from contour lines of the parts and are divided into outline, hole, line and texture. When there is a new class that is not included in the rule base, the system will first retrieve all the possible features from the corresponding image and find out the most similar known class according to the features. The image of the selected known class and the new image are displayed on the same screen. At the same time, the attributes used for the existing class are also displayed. The operator then identifies the difference and decides the new discriminative features of the new class. The addition of new attributes to both new class and existing class is by specifying the type of attributes and the corresponding graphical elements on the screen. In this way, the rule base can be updated by including more and more new classes. The examples shown in the experiments are relatively simply structured object. The limitation of this approach depends on the ability of identifying optimal discriminative attributes when multiple classes and attributes exist.

Luo proposed an automatic rule acquisition system that generalizes the given samples of documents to generate description rules of their layout so that recognition of the document class can be conducted by using the obtained rules[30]. The type of document is limited to all types of forms and tables, which can be easily described by the item blocks and their connections. The structure of forms or tables is grouped by the predefined rules which consist of connection relationship between neighboring item blocks, such as left, right, vertical, horizontal and so on. There are also rules for integration of neighboring blocks into compositive blocks such as n-horizontal connection, n-vertical connection and

so on. For a new type of table or form, a parser will decompose it according to the predefined rules and construct a structure tree with the class of connection in the node and connected members as child or leaf. By this approach, recognition rules can be generated automatically for documents containing a single type of form or table. A table or form with repetitive connection pattern can also be processed by this approach.

Satoh proposed a supporting system for generation of recognition rule used in drawing understanding[31]. This system can assist user in finding the appropriate parameters for recognition. The user specifies the example of recognition target and its classification termed *state*. When there have existed a rule for the recognition target, the system will evaluate the parameters used in the rule and update it in such a way that the new example can be recognized by the result. If not, the system will prompt the user to decide the contents of the rule, i.e. the parameters to be used. The updating of rules is realized by applying Shapiro's Model Inference System, with the extension for handling numerical comparison and types of parameters. It generates a series of conditional test rule that can possibly fit to parameter types of the examples' status, applies all examples to the rule and derives the rule which is consistent with all examples. While the user has to make the effort of identifying suitable parameters, the system can reduces the cost for generation of recognition rules after the parameters are specified.

Application of Digitized Data

There are also researches for making more efficient use of the result of computerized data for other applications. Haralick gives a detailed study on further analyzing the recognition result of mechanical projection drawing for construction of 3 dimensional information[68].

This is an intermediate process between understanding of projection drawings of parts and the application of understanding result. His study makes use of the spatial and geometrical constraints between edges, surfaces and vertices of 3 dimensional objects to find out the correspondence between the edges, surface and vertices in the projection drawings. When the ultimate purpose of understanding is for reusing in CAD application, this is a indispensable process. On the other hand, this process is built with the presumption that all the graphical elements have been correctly recognized/understood. Otherwise, it becomes necessary to have a more complex analysis that goes between the retrieval of graphical elements and reconstruction of the 3-D model. To be more precise, initially retrieve as many graphical elements as possible. Then try to find out the correspondence between the retrieved elements of different projection and the clues for any missing elements that can be retrieved with more definite information obtained by the correspondence hunting.

Lee and Chung suggest a method of reconstructing 3 dimensional information from the recognition result of topographical map drawings[61]. A topographical map drawing describes 3 dimensional information in a 2 dimensional way by depicting the curves on a terrain on which all the points have the same height, and the difference of height between neighboring curves is a constant. It is a good way to use the most common form of presentation for describing topographical information. But since 3-D information is described in 3-D, the result is neither intuitive nor user friendly. When the complete 3-D information is reconstructed, it is possible for other applications to conduct inquiries such as view from different angles, cross-sectional display and so on.

Kasturi and Fernandez present a survey of examples of making use of the information

retrieved from existing paper-based city maps for geometrical information system (GIS)[7]. Automatic name placement on a map drawing is a problem of deciding the order, position and orientation of names printed on a map drawing. This is useful for regeneration of map drawings which used to require a lot of designing efforts for determining the most reasonable way of placing names. A similar problem is generalization of map with smaller scale from maps of larger scale. Original features need to be selected to ensure a constant density of average feature in both the source and target map. In some cases, reduction and combination of features become necessary. Another example is expert system for land-use analysis. After retrieving all the information in a map drawing, a knowledge base is used to perform comprehensive operations such as inquiry of the area belonging to a city, the buildings belonging to a specific institution and displaying a portion of the map as it would appear from a particular vantage point. For the map containing complete road information, inquiry about optimal or all routes between two points, the city along certain route etc. can also be performed. The inquiry can be done by using menu-driven manner or system languages such as

Show [road] type= "all" & within < [state] name = "Washington" >f

2.5 Summary

Inputting of drawings can be roughly divided into preprocessing, feature/symbol recognition and structural understanding. Many researches have been carried out on the fundamental technologies, and the representative ones including their advantages and problems have been reviewed in this chapter. For system construction in general, flexibility and

portability is still subject to improvement. While the reliability of automatic recognition/understanding technologies keeps improving, manual processing by human being is still indispensable and improvement of its efficiency still needs to be explored. In the following sections, the approaches proposed for improving some of the existing problems will be discussed.

Chapter 3

Knowledge Driven Understanding System

3.1 Objective

As stated in Chapter 2, efficient representation of drawing information and inference engine which utilizes them are fundamental technologies in understanding system for drawings. In this study, a paper-based drawing is first converted to digital line information by tracing the contour line of the original drawings and converting the contour lines into approximated vectors within the predefined error range. One of the advantages of this vectorization method is that it can be performed with high speed even with software driven system, no need to mention about its hardware implementation. The *core line* and *glue vector* generated by this vectorization method are then used for representation of the scanned bitmap drawings. The knowledge for understanding of drawings is separated from the inference engine and represented in the format of production rules. The inference engine employs a blackboard model. By emphasizing on the modularity and hierarchical structure of recognition/understanding rules, more efficient and portable understanding systems can be realized. To verify the effectiveness, a prototype system is implemented and applied to

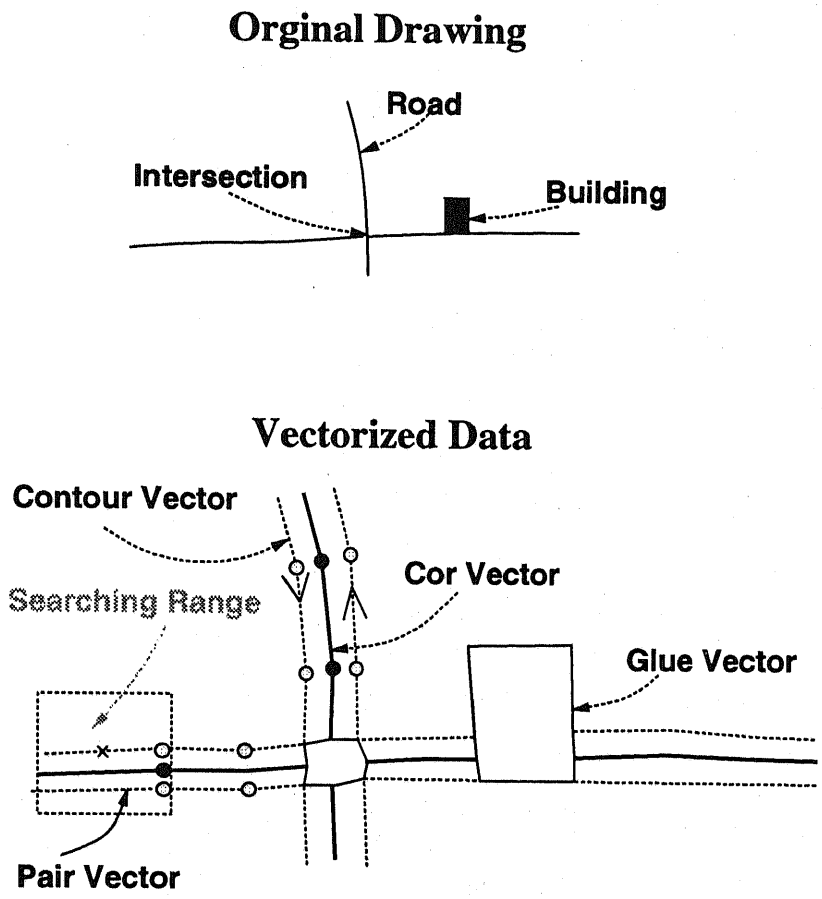


Figure 3.1: Vectorization of line drawing by tracing contour line

several typical drawings[72], [78], [84].

3.2 Vectorization of Line Drawing By Tracing of Contour Lines

In order for computer to recognize/understand all kinds of drawings, it is necessary to convert the paper-based drawings to computerized format, typically CAD line vectors with starting and ending points and the corresponding attributes such as line width, scale etc. Thining is a general and powerful way to convert bitmap images to CAD line vectors. In recent years, many researchs have been conducted for eliminating noisy branches, improving

the quality at intersection of lines. But the computational cost becomes higher. Besides, the filled areas will be thined as their skeletons and lose their original shape information. Therefore the preservation of filled area remains problems. [6],[7], [9], [10], [11],[29]。

In this study, a different approach is adopted for more efficient processing of drawings. First the contour of the lines are traced and approximated with polylines, with the result being called **contour vector**[1]. Since contour vectors along two sides of a line are almost parallel to each other and have opposite directions, the result equivalent to thining can be obtained by generating a line in between such pairs as shown in Fig.3.1. To be more precise, when considering one contour vector, a rectangular searching range is first determined with the center being that of the contour vector and the size being 2 times (decided from experimental result) the width of actual lines. Then compare the angles formed by the other contour vectors within the searching range and that of the contour vector under consideration. If the angle is close to 180 degree, then form a vector, with the start point and end point being the center point of the end points of the contour vector pairs. The connected contour vectors corresponding to a long line will then generate a polyline, and will be called **core vector** hereafter. Since there is no such contour vector pairs at the intersection point or when a filled area touches lines, the core vector becomes open ended. To preserve the information of filled area and the connection information between lines, the contour vectors around the open end of core vectors are linked to the corresponding core vector and stored precisely as part of the vectorization result, called **glue vector** hereafter

[1]To display a line on the screen or evaluate a line, it is necessary to represent them as pairs of starting point and end point called **vectors**. A straight line drawn on paper can be converted to a single such vector or multiple ones when noise or distortion exists.

(Fig.3.1).

By this approach, vectorization of line drawings can be realized at high speed even with software based systems. The information of connection between lines and filled area can be preserved precisely. And the three types of vector data are stored in an efficient spatial structure so that high speed searching of neighbors can be easily performed.

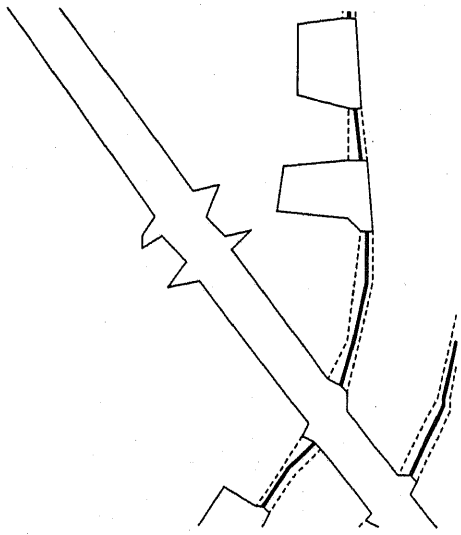
Fig.3.2 shows part of vectorization result for actual map drawings. Dotted lines are contour vectors, thin solid lines are glue vectors and thick solid lines are core vectors. In the shown map drawing, different types of object have different line widths. By choosing the threshold for searching contour vector pairs, core vectors for different type of lines can be easily retrieved. The result shown in Fig.3.2(a) is the result of only vectorizing thin lines which mainly correspond to road lines in the original drawings, (b) is the result of vectorizing thicker lines which mainly correspond to lines of private rail road. This also demonstrates the flexibility of the vectorization approach used for this study.

3.3 Knowledge Representation for Understanding of Drawings

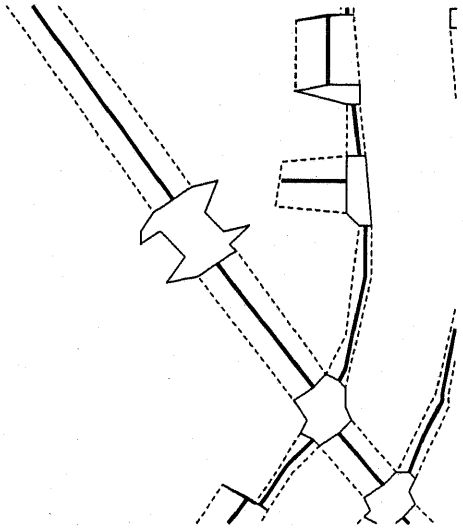
3.3.1 Information in Drawings

Drawings are plotted according to all kinds of regulations and constraints. Therefore, the understanding/recognition of this type of drawings has to be based on the corresponding regulations and constraints. This also implys that the regulations and constaints have to be represented in the form applicable to understanding/recognition systems. Generally speaking, knowledges contained in an drawing can be summarized in Table.3.1.

Basically, drawings are composed of three types of graphical elements: text, line segments



(a)Vectorization of thin line only



(b)Vectorization of thicker line

Figure 3.2: Example of vectorization results for city map drawings

Graphical Elements	Functions	Examples
Text	descriptions about subparts, eg. annotation, explanations	size(digital values), name(special terms), manufacturing process
Line Segments	shape, relationship between subparts, relationship between text and subparts	contour line, connection line center line, dimension line annotation line, abbreviation line
Symbols	type, purpose, manufacture process, procedures etc.	type of electronic parts manufacture procedures of machine parts, type of lands

Table 3.1: Information contained in drawings

and symbols. They are used for describing informations about subparts of drawings, such as relationship between subparts which are determined by corresponding regulations and constraints and hereafter called *domain knowledge* of drawings.

3.3.2 Representation of Knowledge

As stated in Section3.2, the understanding of drawings in this study is based on contour, core and glue vectors. In this way, line segments in a line drawing correspond to core vector and filled area or overlapping symbols glue vectors. On the other hand, the information implied in a drawing and summarized in 3.3.1 is completely different both in abstract level and data format. Therefore, it is necessary to represent the drawing information in a computer readable format, which generally includes geometrical and mathematical features of the graphical elements in drawings. In this study, the following features are being used for representation of recognition targets, i.e. graphical elements in drawings:

1. shape information:

global feature: peripheral length of contour line(L), convex polygon(L_c and circumscribing rectangle(L_r); area of contour line(S), convex polygon(S_c and circumscribing rectangle(S_r) loops.

shape feature: elongation of circumscribing rectangle(R_e), convexity(S/S_c), rectangle likeness(S/S_r),epitomization (L^2/S)

position feature: direction, gravity center of elongation axis.

2. Structural information:

distance: distances between gravity centers of graphical elements in drawings, nearest distance between contour lines.

connection: connection and neighboring relationship between graphical elements (*align, parallel, vertical, neighbor, above, below, to the right, th the left etc.*)

angle: angle formed by graphical elements

Here, graphical elements can be points, straight lines, curves(approximated with polyline) and polygon loops. When features such as direction, circumscribing rectangle and so on are concerned, curve and polygons are represented by their *elongation axis*[16], which is obtained as follows when a polyline or polygon is represented by a list of points $[(x_1, y_1), ..., (x_n, y_n)]$

$$sin2\theta = \frac{b}{\sqrt{b^2 + (a-c)^2}} \tag{3.1}$$

$$cos2\theta = \frac{a-c}{\sqrt{b^2+(a-c)^2}} \tag{3.2}$$

where,

$$a = \sum_{i=1}^{n} x_i^2; \qquad b = 2\sum_{i=1}^{n} x_i y_i; \qquad c = \sum_{i=1}^{n} y_i^2$$

Generally, shape information is used for pattern matching of symbols and text in drawings. Structural information is mainly used for analysis of line vectors, and structure of the object implied in the drawing. The actual matching and analyzing procedures consist of information represented by the above-mentioned features according to the regulations and constraints of target drawings.

3.4 Realization of Knowledge Driven Understanding System

Most of the existing systems are constructed by using many built in knowledges expressed by the information described in 3.3.2. However, because of the variety and quality of drawings, frequent enhancement and modification are necessary. In these cases, systems with built in knowledges will become very inflexible. There have been approaches of systematically organizing the knowledge for recognizing document images[15],[24]. But since the structure of document is easier in the sense that only the type of article blocks and their neighboring relationship are under consideration, graph or tree structure will be enough. Whereas for drawings, it is rather difficult to represent them by these types of simple structures. To solve this problem, this study represents the knowledge used for recognition/understanding with independent knowledge base, which is realized by descriptions of

production rules. A reference engine realized by blackboard model will then interpret the knowledge to conduct the actual recognition/understanding. In this section, the construction of such system is discussed in detail.

3.4.1 Representation of Recognition Knowledge by Production Rules

Production system is one of the most frequently used knowledge representation techniques in the field of artificial intelligence. It is close to human being's way of thinking, and therefore can be easily adopted by recognition and understanding procedures. Recently other knowledge representation techniques have also been reported, but the most fundamental parts are all in production rule format.

A production rule consists of a condition part and an action part. The condition part consists of a set of conjunctive logical predicates, usually related to feature ranges or structural relationships of target elements. The action part consists of a set of commands, usually related to modification of status of elements, addition of new recognition result and so on. When all the conditions of a rule are satisfied, the action part of the rule will be activated and all the commands executed. As a result, the status of certain elements will be changed or new elements be inserted into the processing targets, which will in turn make the conditions of other rules satisfied.

In this study, the production rule bases are built with the following characteristics:

modularity rules are constructed in as independent and general way as possible, so that they will become more compact and easy to understand and modify.

hierarchization Represent complex target in a hierarchical way so that rules will have

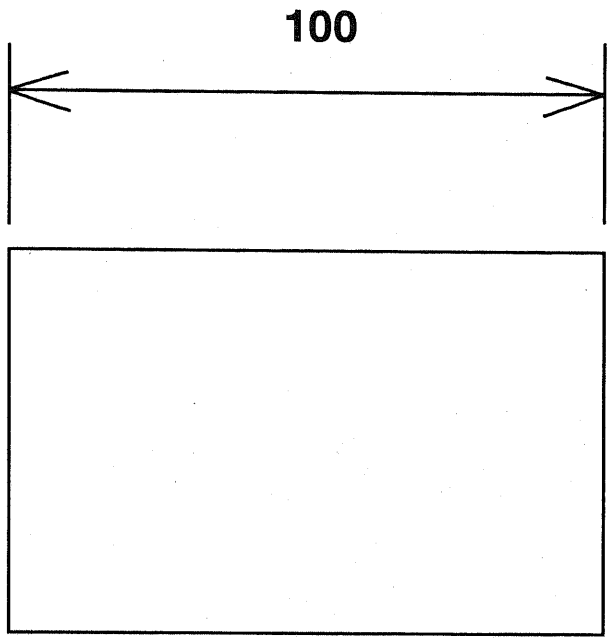


Figure 3.3: Example of dimension description in mechanical drawing

better reusability and distributed and cooperative type of process can be realized.

For example, Fig.3.3 is a simple example of dimension description in mechanical drawings. One type of dimension is for describing the length of parts' edge, or distance between two points, such as center of circle or arc. One common way to do it is to use two parallel lines called *witness line* pointing to the two points to be described. Another line with two arrow heads at both ends and vertical to the witness lines is used to indicate the range of the length or distance. The actual range/distance is located above the dimension line. These regulations can be represented with production rules as follows(literally, not in actual implemented language):

```
IF   line A is not connected
     to other line at end X
```

```
THEN label A as OPEN-X


IF   line A is OPEN-X                AND
     length(A) < length(short)
THEN label A as SHORT-OPEN-X


IF   A is ISOLATED-OBJECT            AND
     area(A) < area(text)
THEN label A as TEXT-CANDIDATE


IF   line A is SHORT-OPEN-X          AND
     line C is connected to A        AND
     angleAC < 45 deg
THEN label A as ARROW-HEAD-CAND


IF   line A is STRAIGHT-LINE         AND
     A is connected to B and C       AND
     B is ARROW-HEAD-CAND            AND
     C is ARROW-HEAD-CAND            AND
THEN label A as ARROW-BODY,
     label B and C as ARROW-HEAD
```

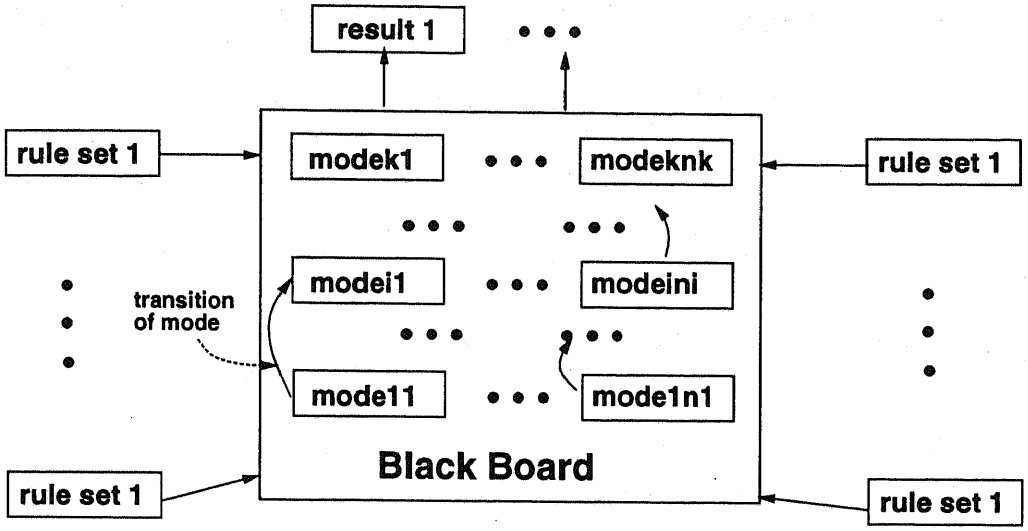


Figure 3.4: Blackboard system

```
IF   A is STRAIT-LINE                AND
     there exists TEXT-CANDIDATE T AND
     T is parallel to A              AND
     distance(A,T) < distance(textToLine)
THEN label A as DIMENSION & ARROW-BODY
     ...
```

When the drawing specifications are different, the corresponding rules will have to be replaced with new ones. Here, since the rules are written with emphasis on modularity, generally only minor modification will be enough. This will be demonstrated in the experiments discussed in Chapter 3.5 through actual examples.

3.4.2 Inference Engine Based on Blackboard Model

Fig.3.4 shows the configuration of blackboard system. Basically, the blackboard model is a structured production system. The graphical elements to be recognized are stored in

the *blackboard* with *mode*s assigned to them. A *rule set* consists of a group of production rules. Each rule set is independently responsible for certain sub-goals, activated by certain modes and performs other mode transitions which may in turn generate new modes for other rule sets. Therefore when parallel processing is supported, the total rule sets realize a distributed and corporative processing. The recognition/understanding system begins with all the graphical elements being of initial modes $mode_{11}, ..., mode_{1n_1}$. Typically they can be the vectorization result. i.e. core, contour and glue vectors; or they can be the result of further retrieval of fundamental graphical features, such as polygon, arc, circle, isolated objects (candidates of text, symbol etc) and so on. The initial modes are altered to new modes whenever a rule set is activated, and new modes will activate new rule sets. The rule activation and mode transition process will go on until no more rule can be activated.

This study makes uses of the blackboard model to realize a distributive and cooperative understanding system for drawings, with rule sets designed both for general purpose and specific purposes. In the following sections, actual rule sets designed for typical drawings will be discussed in detail to show the plausibility and effectiveness of the proposed system configuration.

3.5 Experimental Results

A prototype system for understanding of drawings based on the proposed knowledge driven model is implemented on Sun Sparc workstation. A library of fundamental rule sets and specification/constraint dependent rule sets have been created for three types of typical drawings. Experiments have been conducted by making use of the fundamental rules to

construct knowledge base for three types of typical drawings. By designing the rules with modularity and hierarchization, reusability and flexibility of the existing rules are improved; distributed and corporative processing are also proven to be possible.

3.5.1 Construction of Rules

In the implemented prototype system, a production rule is of the following format:

```
rule(ruleSetName,
     ruleName,
     [condition1,
       ...,
     conditionN],
     [action1,
       ...,
     actionM]).
```

Here, rules of the same rule set have the same *ruleSetName*. $condition_i$ can be a set of disjunctive conditions in the form of $(condition_{i1}; ...; condition_{in})$. A condition can be (a) a procedure such as *thereis(objectName, ObjectID)*, *distance(object1,object2,Distance)* where items beginning with small letters are inputs and those with capital letters are outputs, or (b)a checking condition such as *compare(gt, Value1, Value2)*, *compare(vertical, Line1, Line2)* etc. The system language supported by the prototype system are summarized in Table.3.2.

item	usage	Synopsis
constants	the constants such as threshold , system parameters	*constants(name, Value).* the *name is for target dependent parameters such as maximum/minimum area of text, line width, scanning resolution, error threshold for vectorization, etc.*
thereis	get a graphical element from blackboard	*thereis(name, ObjID).* where *name* can be line, polygon, rectangle, circle, arc etc.; it can also be the name of user defined mode, which is created through *change* below. *thereis(name, [controls], [Output]).* where *control* is the control condition or commands required by *name*; *Output* is the result of the matching action. The element related to the focused element by structural relationship is obtained by this goal.
evaluate	evaluate the attribute value of an object	*evaluate(id, name, Value).* where *id* is the identifier of the object, and can be a list of objects. *name* is the type of value to be evaluated, e.g. length, area, distance etc.
compare	compare two values by the specified key	*compare(key, Value1, Value2).* where *key* can be *gt, sm, equal etc.*; *Value1* and *Value2* can either be constants or expressions such as 2*Width.
addBuffer	set the buffer for intermediate result	*addBuffer(bufferName, bufferContents).*
getBuffer	get the contents of buffer for intermediate result	*getBuffer(bufferName, BufferContents).*
updateBuffer	update the contents of buffer for intermediate result	*updateBuffer(bufferName, BufferContents).*
change	change the mode of an graphical element in the blackboard or the mode of inference	*change(id, attributeName, attributeValue).* and *change(mode, NewMode)*

Table 3.2: Language for rule construction supported by the prototype system

3.5.2 Fundamental Rules and Procedures

To facilitate the rule construction, a library of fundamental rules and procedures is prepared. Here procedures are for evaluation of attribute values or relationships such as area, length, distance etc. The total number is about 100 in total. The contents can be divided to two categories:

1. evaluation of **shape information**: mainly used for retrieval of information for initial stage of recognition, such as symbols consisting of only one single polygon, parts of a complex symbol, broken part of an object etc.

2. evaluation of **structural information**: for describing the most fundmental spatial relationships among objects.

Some examples of them are shown in Table.3.3, where a polyline is represented as a list of points $[(x_1, y_1), \cdots, (x_n, y_n)]$, a line a vector $[(x_s, y_s), (x_e, y_e)]$. Currently, the total number is about 60. These rules are not guaranteed to be exhaustive, but just some examples of how the fundamental drawing information can be implemented in rule form; they can be easily expanded or combined to form other special purposed rules, either designed by user as customized rules or added by programmer as new system rules.

There are also operations over the blackboard, such as element searching within a specified range, mode transition of an element and so on. For operations on internal data base, data structure, there are about 60 rules which are mostly hidden from end user. A *message(Massage)* command, the result of which being always true, is implemented for debugging purpose, where *Massage* can be a text string or a list of intermediate variables.

Rule Set	Graphical Elements	Contents
shape	area	$0.5 \mid \sum_i((x_i - x_{i+1}) \times (y_i - y_{i+1})) \mid$
	convex area	area of convex hull
	length of line L	$\sqrt{(x_e - x_s)^2 + (y_e - y_s)^2}$
	horizontal circumscribed	$x_{min} = min_i(x_i), y_{min} = min_i(y_i)$
	rectangle (CR)	$x_{max} = max_i(x_i), y_{max} = max_i(y_i)$
	height of CR	$min((x_{max} - x_{min}), (y_{max} - y_{min}))$
	width of CR	$max((x_{max} - x_{min}), (y_{max} - y_{min}))$
	elongation axis	equation (3.1) or equation (3.2)
	elongation rectangle(ER)	the CR obtained along the elongation axis
	height of ER	$min((x_{max} - x_{min}), (y_{max} - y_{min}))$
	width of ER	$max((x_{max} - x_{min}), (y_{max} - y_{min}))$
	convexity	area / convex area
	rectangle-likeness	area / area of ER
structure	angle between lines	$acos((\Delta x^{\{1\}}\Delta x^{\{2\}} + \Delta y^{\{1\}}\Delta y^{\{2\}})/(L^{\{1\}}L^{\{2\}}))$
	angle between line and polygon	angle between line and elongation axis
	angle between polygons	angle between elongation axes of the two polygons
	parallel lines	two lines with angle between them smaller than Θ_p
	parallel polygons	two polygons with angle between their elongation axes smaller than Θ_p
	aligned lines	parallel to each other AND within the range of line width
	aligned polygons	parallel to each other AND within the range of W_{pp}
	above	$min(y_i^{\{1\}}) > max(y_i^{\{2\}})$
	vertical lines	two lines with angle between $90 \pm \Theta_v$
	connected lines	two lines linked by *glue* vectors
	connected polygons	two polygons linked by *glue* vectors
	nearby lines	lines that intersect with the searching range centered at the center of the line and with width W_s and height H_s
	cross intersection	four lines connected to each other AND form two pairs of *aligned* lines AND the two pairs are *vertical* to each other
	T intersection	three lines connected to each other AND two of them are *aligned* AND the rest one vertical to the other two

Table 3.3: Examples of fundamental rule/procedure sets

Figure 3.5: An example of vectorized city map drawing

3.5.3 Understanding of Layout Drawings

Layout drawings are mainly for description of shapes, positions of abstract objects and so on. All kinds of metropolitan maps (such as city map, topographical map, contour map etc.), meteorological map and so on are examples of layout drawings. Among all types of layout drawings, city map drawings have most of the representative features. In this section, a prototype system for recognition/understanding of city map drawings is discussed. The system trys to automatically recognize as many graphical elements as possible and leaves those rejected by the designed rules for interactive recognition process[78],[74].

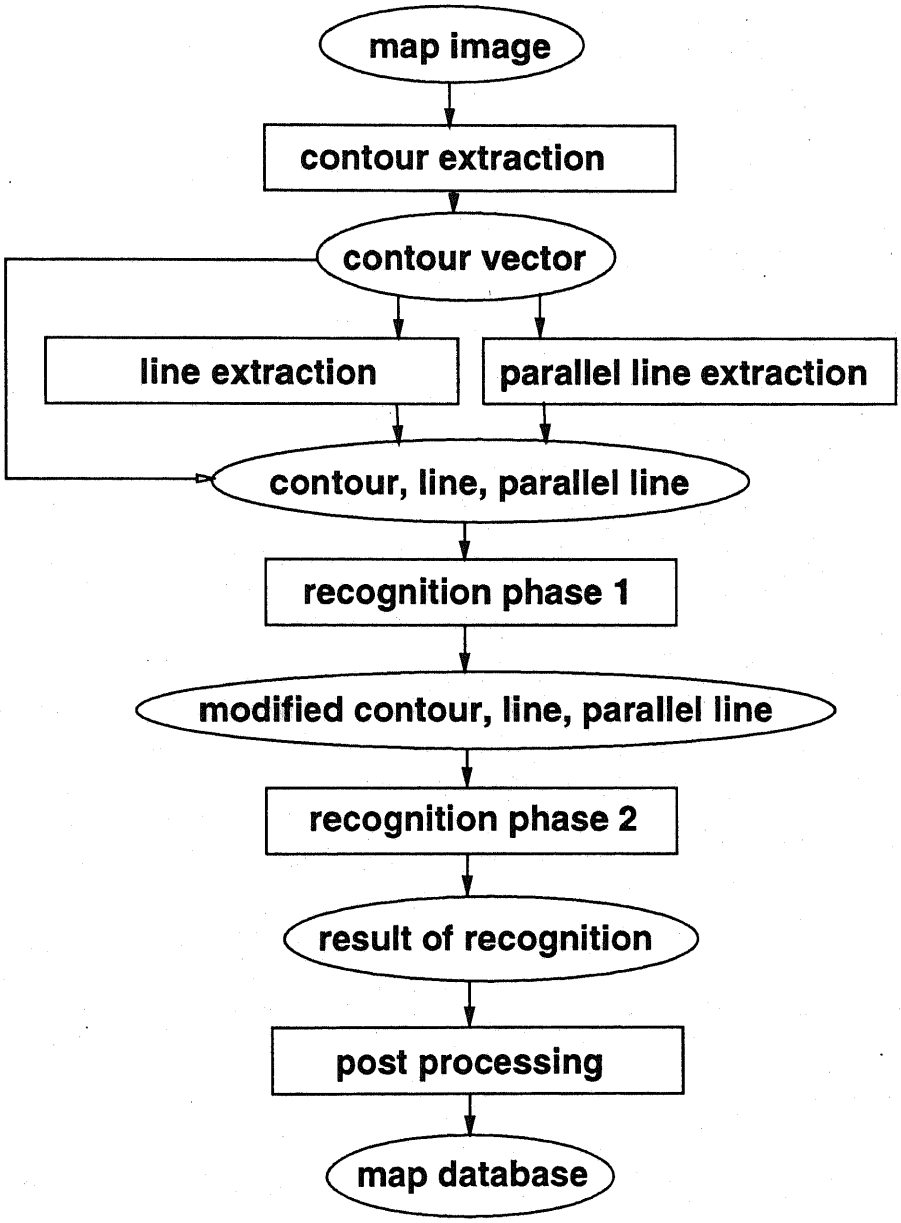


Figure 3.6: Recognition flow of city map drawings

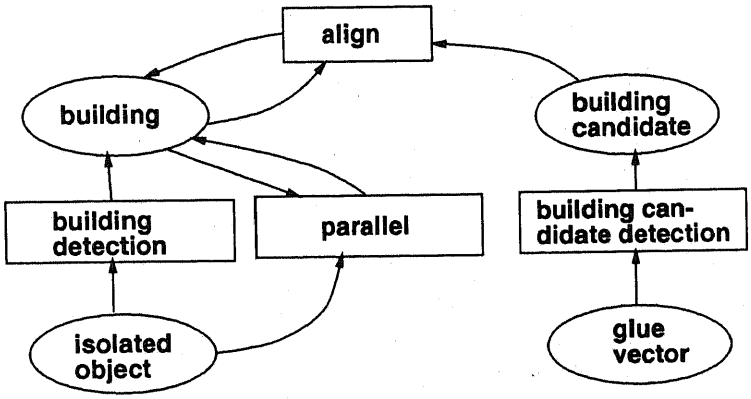


Figure 3.7: An example of rules for building symbols

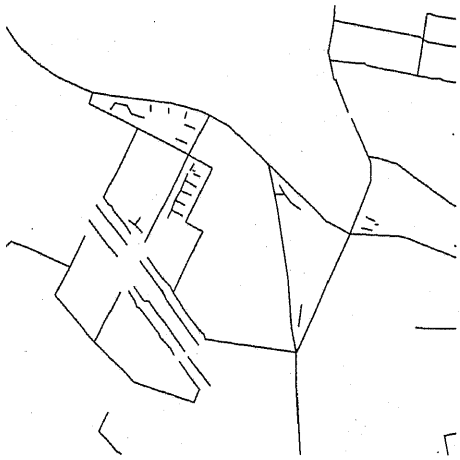
Fig.3.5 shows an example of vectorized city map drawing. The main purpose of recognizing this type of drawings is to retrieve graphical objects such as road, rail road(JR and private rail road), hatching area, building and other symbols. Fig.3.6 shows the flow of the recognition process for city map drawings. "recognition phase1" retrieves graphical object with comparatively simple and explicit patterns such as rail road, hatching and building. The "recognition phase 2" retrieves road lines which is comparatively fuzzy when the density of graphical objects is high.

Besides the fundamental rules stated in Chapter 3.5.2, There are a total of 21 specification dependent rules, written with the combination of the fundamental ones. There are also 10 constants that are derived from the specification of city map drawings and used for recognition of the corresponding graphical symbols. The rules are mainly divided into two sets: one for simple shape recognition and the other for more complexed situations where the result of the first set becomes necessary. Fig.3.7 shows an example of a sub-rule set which belongs to the second rule set and is used for recognization of building symbols which overlap with road lines. Items in ellipse frames are modes of graphcal elements; items in

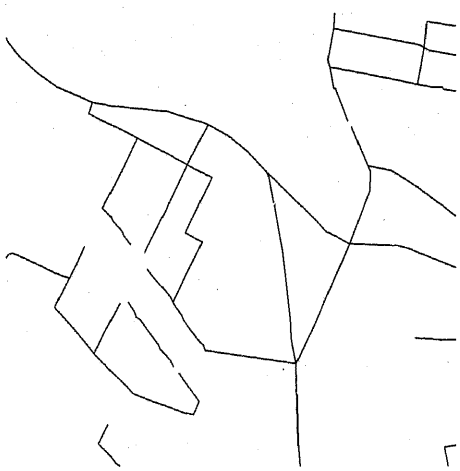
rectangle frames are rules. There are a total of 7 rules which make use of restrictions and interelations to recognize building symbol, in addition to the ones that make use of shape parameters only, as in the case of isolated symbols. The polygons formed by glue vectors mainly contains: (a)building symbols overlapping with road lines, (b) rail road symbols, (c)intersection point of roads, (d)building symbols vectorized because of distortions, and (e)others. The glue vectors for ideal patterns can be classified by their shape parameters and their surrounding information. In the case of building symbols, they can be classified in two steps. The first step is to match the symbols with shapes close to ideal patterns, i.e. similar area, convexity, rectangle-likeness and so on. Here *similar* means that the corresponding values are different by a certain threshold, typically 5% in the prototype system. The second step is to find other symbols with more distortions, they can be retrieved as building candidates by a more flexible threshold. Then the adjacency with known building symbols and the alignment will be checked since most of the buildings have the same direction of outline.

Fig.3.8 shows examples of road recognition by two rule sets. Fig.3.8(a) shows part of the original data, (b) the result generated by the rules in the first rule set stated above, i.e. recognizing the road lines drawn under simple environment. The result shown here contains some non-road lines, which are generated mainly by railroad lines parallel to road line and large building symbols, also parallel to road line. Fig.3.8(c) shows the result obtained by activation of sub-rule sets for roads in the second main rule set. By this second rule set, the result of existing road is further checked against their vicinities to get rid of the non-road lines. Similarly, Fig.3.9 shows the final result of part of overlapped building symbol

(a) Original image



(b) Result of individual road recognition



(c) Result of road recognition with environmental information

Figure 3.8: Example of road information in different stage

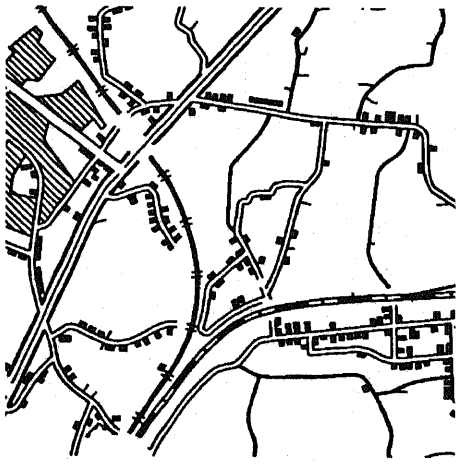
recognition.

For actual drawings, most of the digitized glue vectors can be classified by the above mentioned rules when the vectorization distortion is not severe. But the rest are difficult to classify by explicit productions and have to be processed through interactive processes with the help of human being. The interactive process is usually tedious and time consuming but at the same time indispensable in the existing drawing understanding systems. This study has proposed more efficient algorithms for interactive systems and they will be discussed in detail in Chapter 4

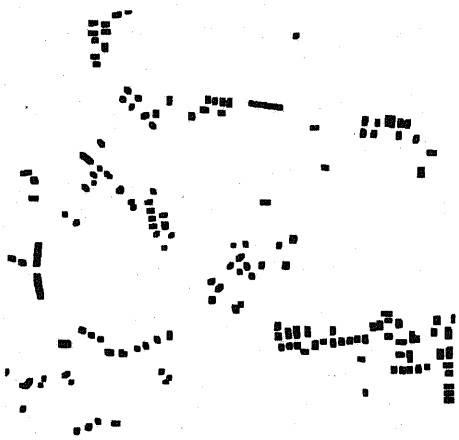
3.5.4 Understanding of Connection Drawings

Connection drawings are mainly for description of the type of objects and the relationship between objects. Digital circuit drawing, printed circuit drawing, wiring drawing, and all types of utility drawings are some examples of connection drawings. In this secton, one type of layout drawing, i.e. pipe line layout drawing, is taken as the target of understanding. Fig.3.11 shows one such type of drawing which is one of the actual design drawings for Tangshan City, China, and manually drawn. For efficient management of the existing pipe layout data, it is required to input all the information into a geometrical information database. Currently, one such drawing will take a veteran worker two working days to input and missing item or mistake tend to occur frequently. At this pace, the total drawings will take several staff years to complete.

A prototype system has been built for inputing this type of drawings more efficiently. The initial aim is to locate all the pipe lines in the drawing, so that later on other information such as length of each sub-part of pipe, gradient and so on can be added. Fig.3.10 shows the



(a) Original image



(b) recognition result of building overlapped symbols

Figure 3.9: Example of building symbol recognition

graphical elements in a drawing, their relationships and the procedures necessary necessary for their retrieval. Some of the rules used in the procedures are listed in Table.3.4. Basically the rules are divided into two major groups as indicated by the dotted line in Fig.3.10. The first major group is for retrieval of fundamental elements that either are used for forming complex object or forming description for compound information. The main rule sets are line tracing for pipe line candidates, range searching for symbol candidates and detection of dimension candidates. The second major group is for retrieval of structural information, which consists of rule sets such as structural symbol detection, determination of type of pipe and connection between dimension set and pipe lines. Since most of the symbols have a circumscribing circle and some structured internal elements, the corresponding recognition requires time consuming techniques such as art fitting or template matching. To reduce the computation cost and improve the reliability, the retrieval of such symbols are based on the result of retrieving candidates of pipe lines. This is because that (a) the symbols are drawn on the pipe lines and connect two pipe lines together, (b) the candidate of pipe lines can be traced comparatively with ease. Therefore, the candidate of symbols can be limited to those between pipe line candidates. Currently, besides the fundamental rules as have been used in Section 3.5.3, there are a total of 28 specification dependent rules.

Fig.3.12 shows the result of recognizing the pipe symbols and classifying the type of pipes. The symbols are shown with filled circles, and the attributes of retrieved pipes are displayed with different colors: black for rain , blue for tap water, brown for heat and cyan for sewage. Since the understanding of dimension lines is similar to that of mechanical drawings, the analysis of which will be discussed in the following section.

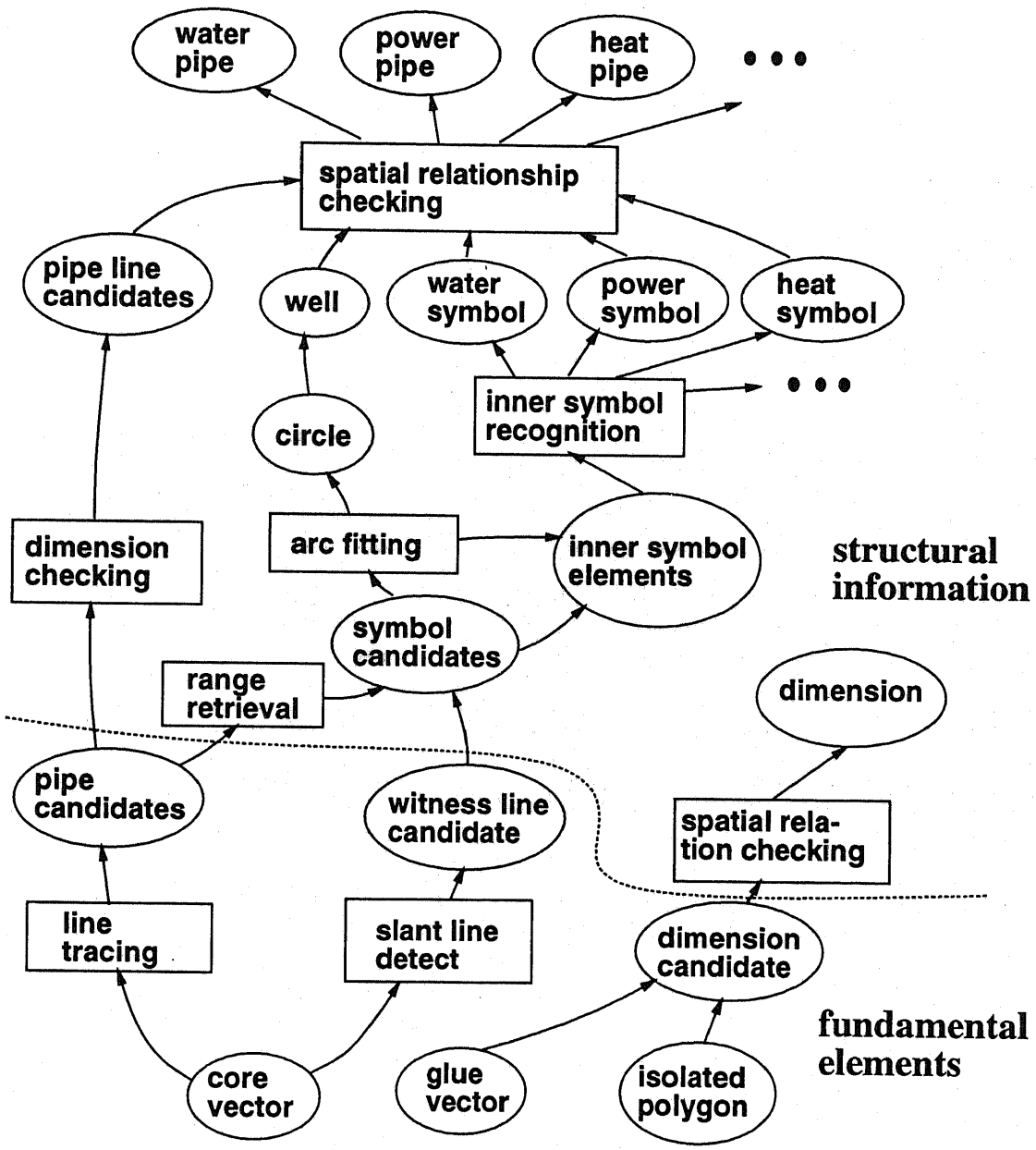


Figure 3.10: Graphical elements and their relationship in pipe line layout drawing

Rule Set	*Graphical Elements*	*Contents*
symbol	well	has circle AND connected to at least two lines
	water	has circle AND has "upper" character inside
	sewer	has circle AND has "down" character inside
	ash	has circle AND has "H" character inside
	gas 1	has circle AND has three parallel straight lines inside
	power	has circle AND has one straight line inside
	valve	two triangle like polygons connected together
	text	isolated polygon AND has similar neighbors
	dimension	text AND parallel to a nearby line
structure	dimension line	open ended line AND connected with slant line
	pipe candidate	straight line AND can be traced down until open ended
	water pipe	pipe candidate AND (has *water* symbol OR is connected to water pipe through a *well* symbol)
	road	straight line AND connected to curve AND no symbols on it

Table 3.4: Examples of rules for understanding of pipe line layout drawings

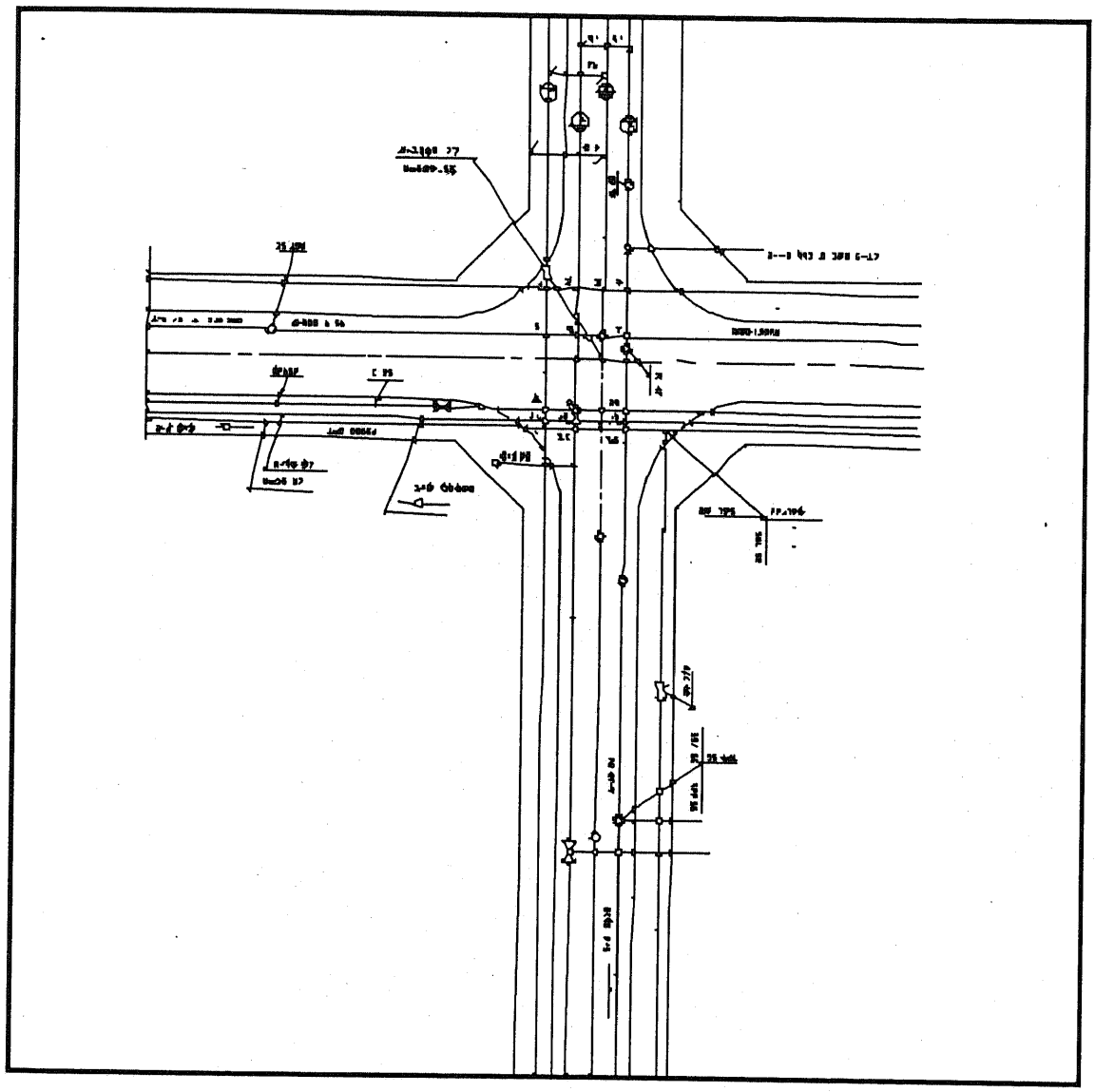


Figure 3.11: An example of vectorized pipe line layout drawings

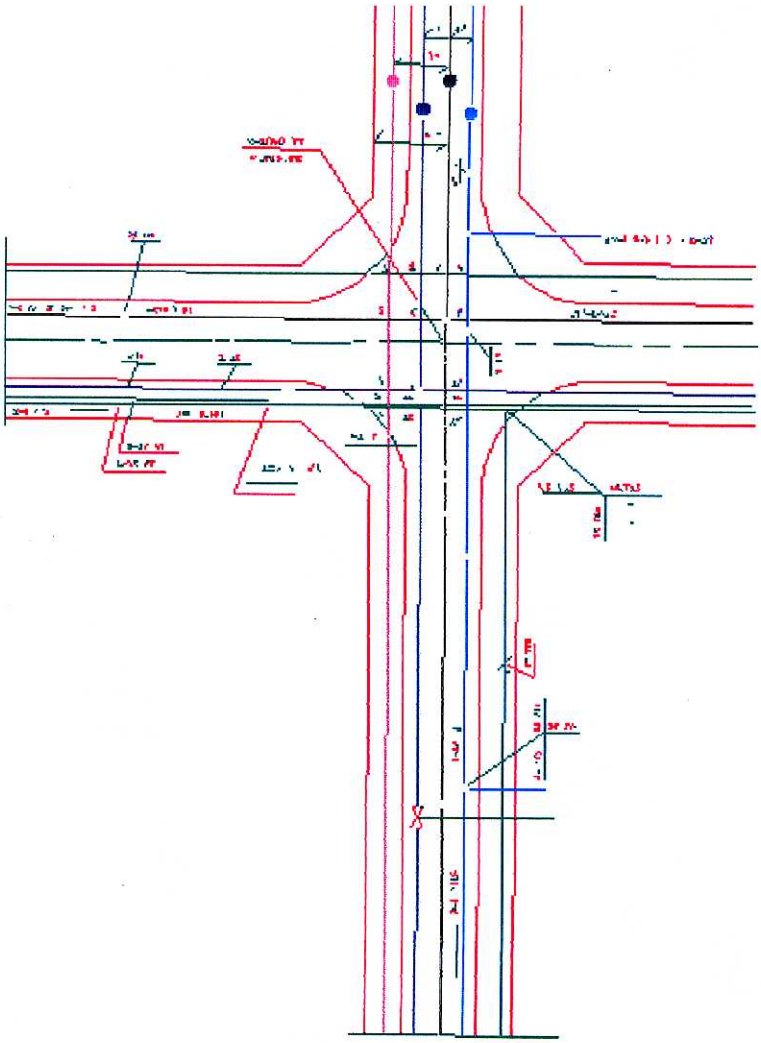


Figure 3.12: Understanding result of pipe line layout drawings

3.5.5 Understanding of Mechanical Drawings

Mechanical drawings are for description of information about mechanical parts. The information includes shape, structure, manufacturing procedures and so on. Plate parts drawing, soldering procedure drawing, orthogonal drawing and so on are examples of mechanical drawings. Here, plate parts drawings have the following characteristics:

- contains only one projection drawing
- describes shape and structure information of parts
- contains all basic graphical elements such as line segments (solid line, dotted line, center line, radius line etc), symbols(arrows, manufacturing symbols, text etc) and so on

Therefore, plate parts drawing can be considered representative of mechanical drawings. In this section, plate parts drawing is taken as the target of experiments.

Fig.3.15 shows an example of plate parts drawing. The typical graphical elements in a plate parts drawings and the relationships between the elements are shown in Fig.3.14. Basically, the graphical elements are closely related to each other, meaning that most of them depend on or explain one or more other elements. The relationship between graphical elements has clear hierarchy at the bottom part corresponding to symbols, and complex interlinkage at the upper part corresponding to structural elements. The rule sets are thus constructed with two major groups: symbol retrieval and structure analysis. The symbol retrieval group mainly performs pattern matching with the geographical features such as shape and position to recognize arrow head, arrow, text, text string, symbol and

so on. The structural analysis part mainly performs retrieval of structural information and consequently retrieval of structured objects such as all types of dimension lines.

Table.3.5 shows some examples of rules supported by the prototype system. They are categorized by their related graphical elements and mainly effective for ideal patterns or structures. There are also some rule sets that are designed for more noisy situations. For example, the arrow symbol sometimes are distorted because of blur of the original drawing, so that rules designed for ideal situations can not retrieve then, hence the dimension line. On the other hand, since there will be a text string above dimension line and parallel to it, a line can be assumed to be a dimension line if the above condition can be satisfied, and there is no other lines closer to the corresponding text string. In the prototype system, about 20 such rules divided into 5 groups are prepared as supplement to the rule sets designed for ideal situations. Fig.3.13 shows an example of elements that can contribute to recognition of dimension line for radius.

Fig.3.16 shows the recognition result generated by the implemented prototype system. Each part of the drawings are correctly recognized/understood and fairly reprinted. The lines printed in yellow are witness lines; lines printed in orange are dimension lines; lines printed in green are radius lines; lines printed in white are outlines. The dimensions of every edge of the parts are calculated according to the witness lines and length of radius. The radius of arc are inputed manually since currently there is no character recognition module in the system. Some of the dimension lines do not have perfect arrow heads and some of the short dimension lines are corrupted by blur in the original drawings. These special situations are all covered by the extra rule sets stated above.

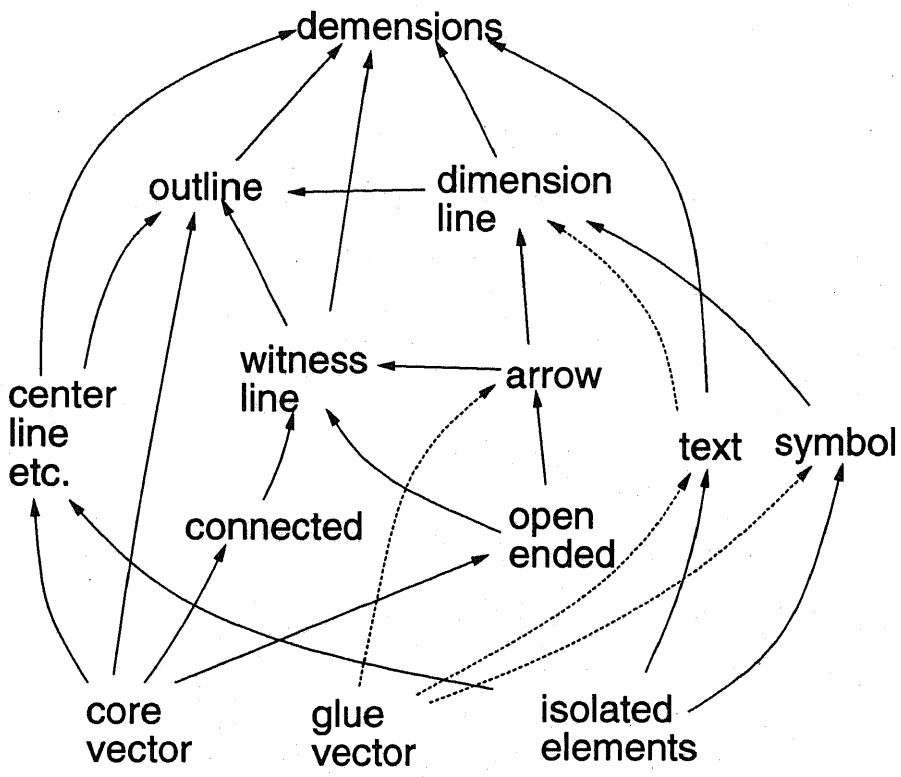


Figure 3.13: Graphical elements in mechanical drawings

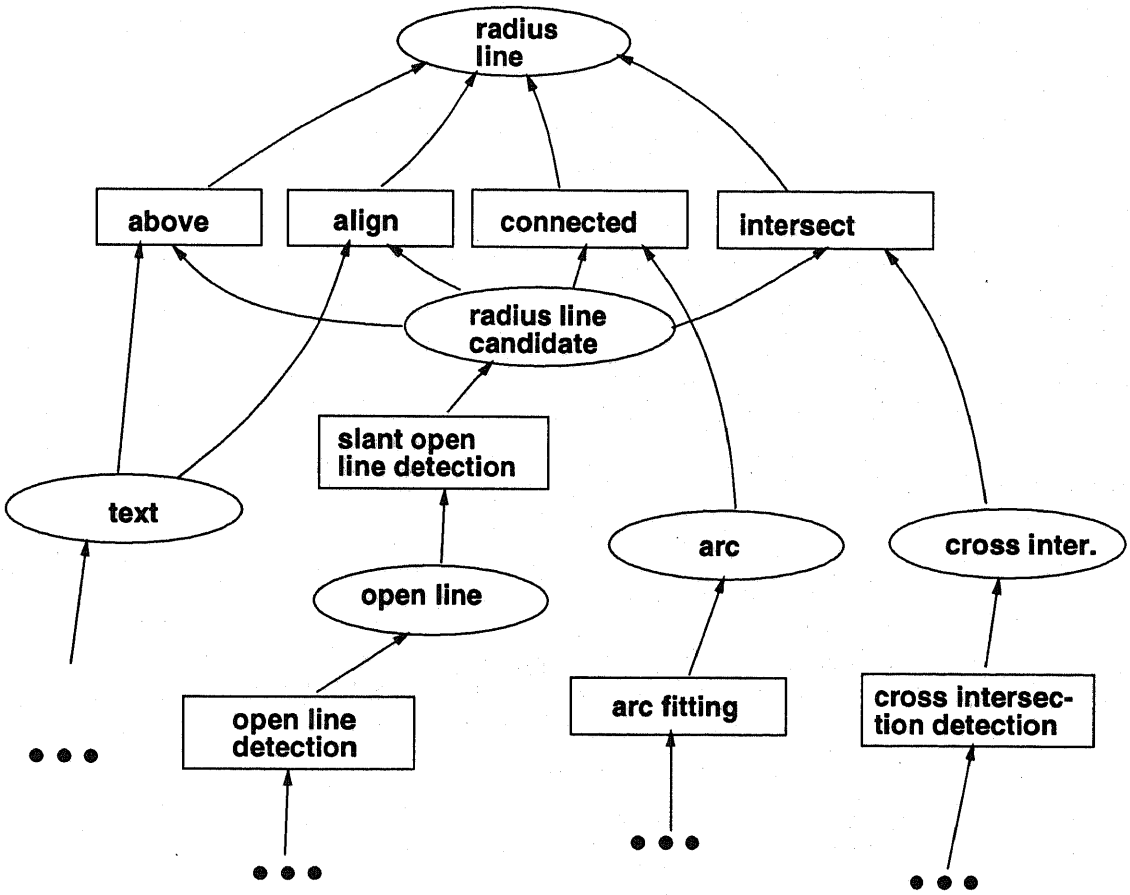


Figure 3.14: Elements that can contribute to recognition of radius line

Rule Set	Graphical Elements	Contents
symbol	character string	short open line and angle < 45
	arrow head	short open line and angle < 45
	arrow	two arrow heads and one arrow body
	arc candidate(1)	short connected line AND with arrow side of radius line pointed at
	arc candidate(2)	distance to circle/arc candidate $< D$
witness	type 1	open at both end
	type 2	vertical to dimension line
	type 3	slant AND intersect with other line at one end AND connected to dimension line at the other end
dimension line	type 1	with arrow head at both end
	type 2	with arrow head at one end AND open at the other end
	type 3	aligned with text at one end
	type 4	parallel with text string AND (I) below the text string
	type 5	parallel with text string AND to the left of text string
	type 6	connected to a dot at cross intersection
circle or arc	type 1	slant AND with arrow at one end
	type 2	slant AND with text aligned
	type 3	slant AND with text above
	type 4	slant AND with arrow at both end
	type 5	slant AND with arrow at both end AND with text above
	type 6	Z type AND with arrow at one end AND with text aligned at the other end
	type 7	type 1 AND connected to curve candidate
	type 8	type 1 AND pointing to line vertical to it
position	arc center	intersection of straight line with radius candidate
	circle center(1)	cross intersection with diameter line passing through
	circle center(2)	cross intersection with arc line ending

Table 3.5: Examples of rules sets for plate parts drawings

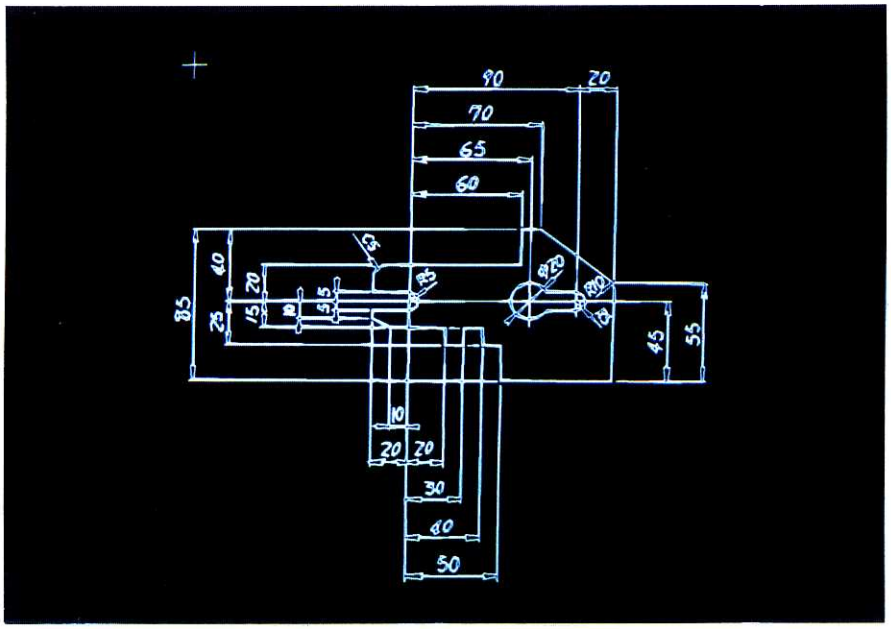


Figure 3.15: An example of plate parts drawings

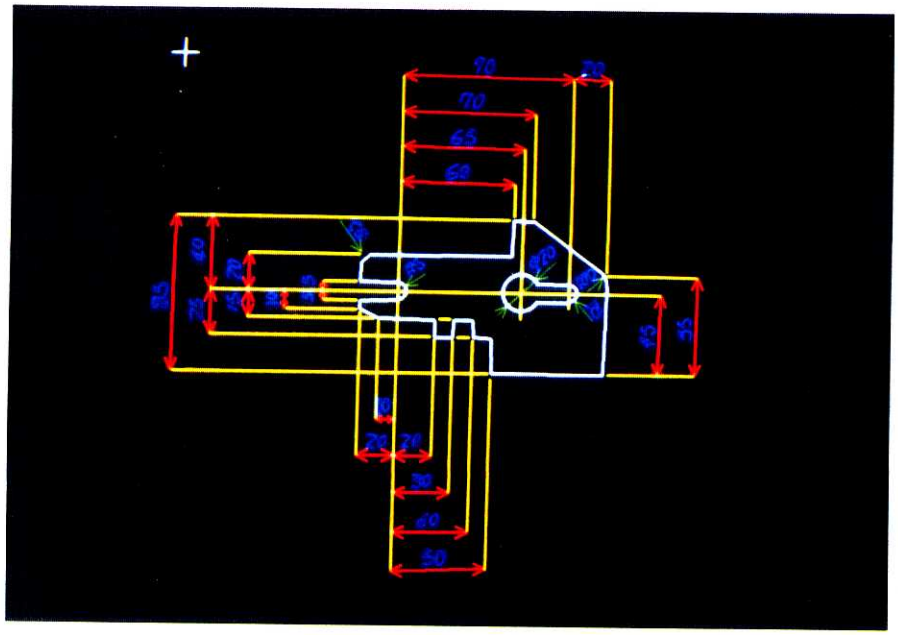


Figure 3.16: Result of understanding plate parts drawings

3.5.6 Evaluation of the Prototype System

The performance of the prototype system is evaluated in two aspects: ease of system construction and efficiency and portability of rule construction. By separating the knowledge used in recognition and understanding of drawings from inputting system, the representation of knowledge becomes more compact and explicit when compared with systems implemented by combining the processing knowledge with the processing procedures. Since the current system takes the production rules as input and interprets them directly, there is no need for complex compilation or format conversion and the task of building a recognition system becomes only arranging or combining the structure of rules/rule-sets. The maintenance of the total system also becomes easier, since most of the maintenance tasks only deal with modification or expansion of knowledge. When debugging is necessary, the *message* command has been proven to be able to fulfill most of the error isolation tasks. While more concrete evaluations (such as construction time, trial-and-error time and so on) depend on many uncertain factors,(such as programmer's efficiency of doing work, computer's environment etc.), the above discussion can draw the conclusion that system construction and maintenance become easier by the prototype system than by built-in type systems.

The composition of the rules designed for the three types of drawings is summarized in Table.3.6. The *shared* item indicates the number of specification dependent rules shared by different type of rules. All the three types of drawings only need to write from 30% to 50% of specification dependent rules. The rest can be covered by the fundmental rule/procedure library. The shared rules are either in a direct way or through adjustment of *constant* database. Therefore, the prototype system can also be said to be efficient and have certain

Drawing	Total	Fundamental	Spec. Dependent	Shared
city map	65	42	21	2
pipe line	82	50	22	10
mechanical	120	60	48	12

Table 3.6: Composition of rules for three types of drawings

degree of potability in rule construction.

3.6 Summary

In this chapter, a more flexible way of system construction for inputting drawings have been proposed. For improvement of efficiency and portability of system construction, the recognition knowledge is represented by production rule base and separated from the recognition kernel. The rule base drives a blackboard system to perform the actual inputting process. In this way, the system is proven to have the following advantages:

1. easy construction of system: by resuing the fundamental and existing rules, less effort is required for constructing new system.

2. easy maintenance of system: new specifications can be catered for by only changing/modifying the corresponding rule sets.

3. easier for designer to concentrate on efficient system design: since production rules are close to human being's way of thinking, more efficient design can be easily reflected in the system construction.

Recently, other approaches have also been proposed for the same purpose. One example is the state transition system which can realize a general purpose system for various target

drawings[28]. While taking more time to get used to the specific definition language, the system can support top down inference. In fact, the basic transition rules in this system are also in production rule format. Therefore, the production rule constructed by the system proposed in this chapter can be easily ported to the state transition system when more top down inference is necessary. Another example is the one discussed in references [82], [86], and [87], where recognition procedures are described in object-oriented matching tree format, so that resuability and portability can be realized. Again, while the structure of the system is more complex and therefore takes longer time to get used to, the matching tree is also based on production rules. Therefore, the knowledge driven system proposed in this chapter can also be a good fundamental platform for other approaches because of its simplicity and transparency.

Chapter 4

Interactive Recognition System with Learning Ability

4.1 Objective

By the automated understanding system stated in Chapter 3, construction, modification and enhancement of system become easier. When there are new types of specifications, noises or distortions, the system only needs to make changes in the corresponding production rules. But there are still graphical elements that are irregularly distorted by all types of reasons and can not be covered by the limited explicit rules. This also happens to any other existing approaches. There are researches on getting rid of the influence of noises or distortions, but normally they result in more complex, target dependent and time consuming algorithms and still cannot ensure perfect recognition/understanding[49], [51]. As a result, manual operations have to be introduced to correct the wrong recognitions and complete processing of the rejected ones. From the statistics of the existing systems, manual operation usually takes much more time than the automated process. Besides, since the manual process is tedious and boring, reliability tends to be unstable.

This chapter proposes a more efficient way of system construction to improve the relia-

bility of recognition and the efficiency of manual operations. The principal policy of the proposed system construction is to automatically process graphical elements having little noises or distortions by strictly conditioned rules. For the rest of elements, use more flexible rules to propose multiple possible recognition results. An operator will complete the final judgement for the proposals. The processing procedures for the possible results are implemented beforehand so that whenever the operator selects a proposal by clicking on the mouse buttons, the system will complete the rest of processing. In this way, manual operation can be realized by mouse clicking for most of the time. Only when all the proposals fail are the manual operations necessary. When the correct rate of proposals is high, the operator only needs to click on the "yes" button for most of the time, which can be done much faster and more reliably than moving the cursor around to select the correct answer. Therefore, to improve the correct rate of proposals, the generation of proposals is realized by making use of the processed elements. Further more, to maintain the correct rate of proposal at a steady rise during the interactive processing, a planning algorithm is also proposed for determining the order in which the unknown graphical objects are processed. The experiments using actual drawings show that the proposed algorithms are effective for the objectives. The planning algorithm can also be applied to other fields such as training of neural networks. [80],[74], [81],[73],[76].

4.2 System Configuration

Fig.4.1 shows the configuration of the proposed interactive recognition system. First, the system digitizes the paper-based drawings to computer graphical elements, particularly line

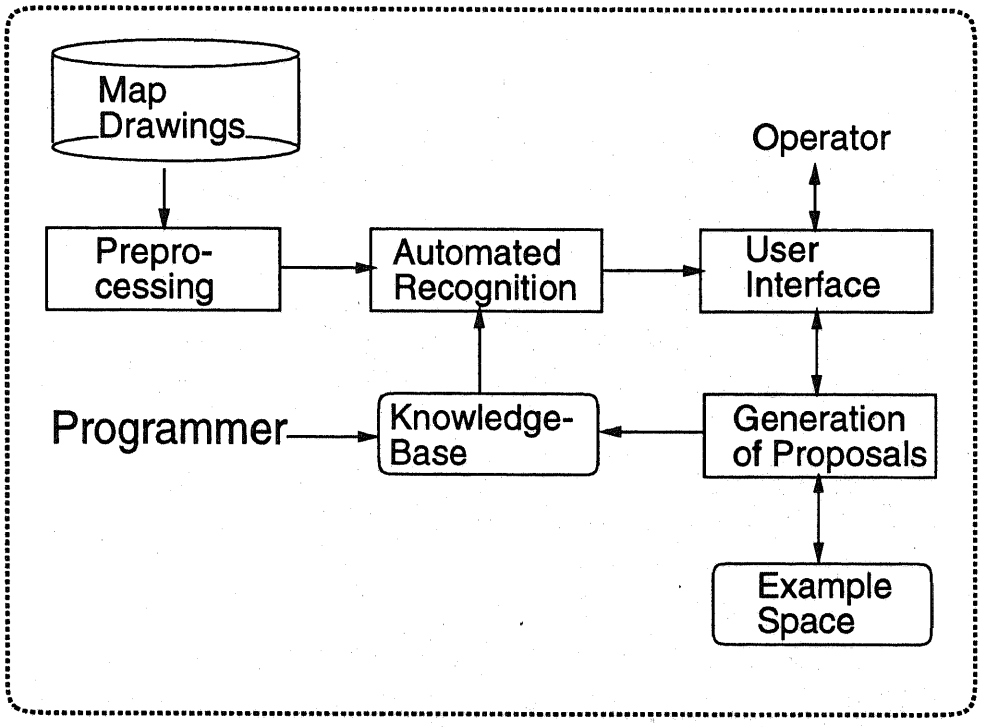


Figure 4.1: Configuration of the proposed interactive recognition system

segment data for drawings, by *Preprocessing* module. The line segment data represents line elements in the original drawings with their skeleton lines in the form of start point and end point pairs called *core vector* as stated in Chapter 3, filled areas with their contour lines called *glue vector*. In this way, the information of filled area can be retained while digitizing line drawings, even when the filled area overlaps with or touches lines. The digitized data is then recognized by the *Automated Recognition* module, which is driven by the *Knowledge Base* consisting of recognition rules.

After the automated recognition process, the remaining unprocessed graphical elements are passed to *User Interface* module. By this module, unprocessed elements are presented to operator together with the proposed recognition results generated by the *Generation of Proposals* module. The system continues to process the rest of unknown elements when

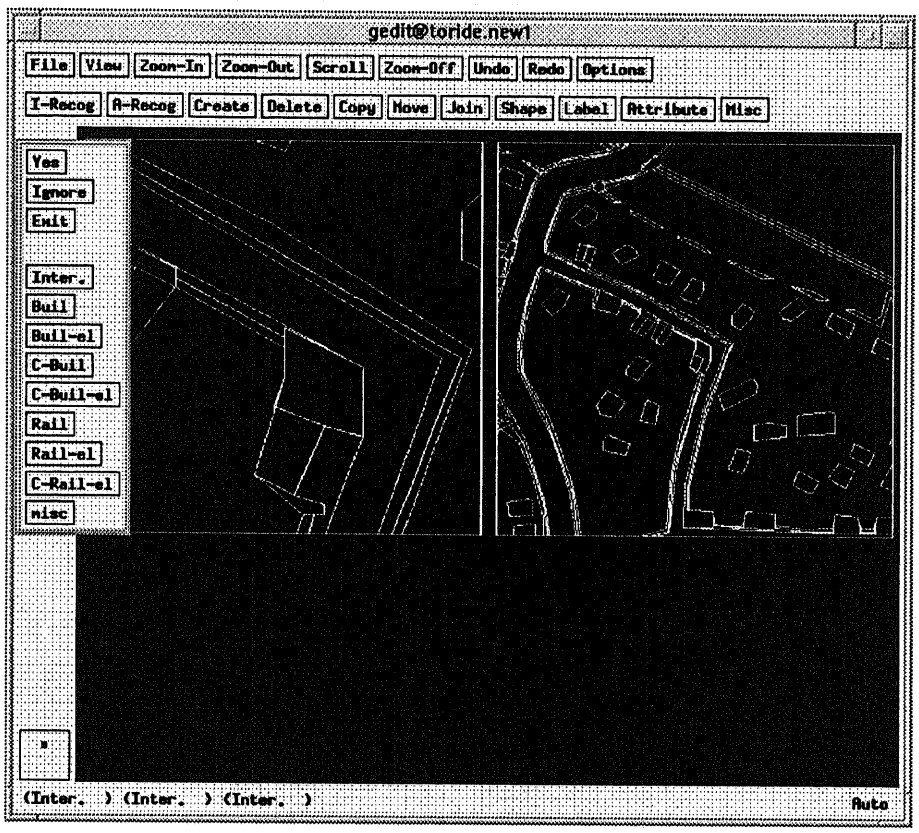


Figure 4.2: Layout of user interface for interactive recognition

the recognition proposals are confirmed by operator. The processed elements and their recognition results processed in this module will be stored in the *Example Space* and used by *Generation of proposals* module to generate proposals for remaining elements.

The screen layout of user interface system is shown in Fig.4.2. The element to be recognized is displayed at the two largest windows in the center, with different scales – larger scale for clearer view and smaller scale for environmental information. The proposals are displayed at the bottom of the main window. At the left side of the main window, there is a matrix of buttons for confirmation of proposals. The "Yes" button is for confirming a correct proposal, other buttons are for inputting the correct answer when the proposal is

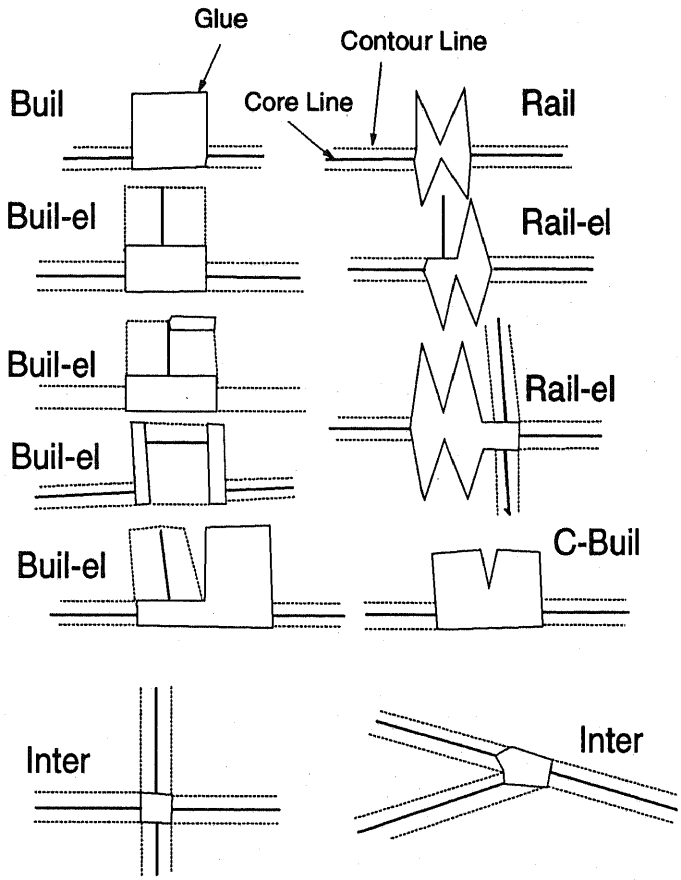


Figure 4.3: Example of distorted glue data

wrong. When the proposals are always correct, there is no need to move the cursor around and select the correct button. In this case, processing speed becomes faster. There are also pull down menus for manual operation when all the proposals fail.

4.3 Description of Target Objects

The processing target is chosen as the glue vectors in Fig.3.5. As stated in Chapter 3, when a symbol overlaps with lines, the original shape of the symbol will be preserved by storing the corresponding glue vector. When there are little noises or distortions, this type of glue vectors can be recognized. But for other cases, automated process becomes difficult.

Therefore manual operation becomes necessary. Fig.4.3 shows some typical glue vectors and their typical distortions which are difficult to be covered by explicit production rules.

In this interactive recognition system, the graphical elements are described by the following attributes:

core vector :

- shape: width of line, length
- connection: number of core vectors connected;

  the average, variance, maximum value and minimum value of the angles between the core vectors

  the average, variance, maximum value and minimum value of the lengths of the core vectors

  the average, variance, maximum value and minimum value of the widths of the core vectors

glue vector :

- shape: area, peripheral length, epitomization, convexity, elongation, number of projection.
- connection: number of core vectors connected;

  the average, variance, maximum value and minimum value of the angles between the core vectors

  the average, variance, maximum value and minimum value of the lengths of the core vectors

  the average, variance, maximum value and minimum value of the widths of the core vectors

isolated polygon
- shape: area, peripheral length, epitomization, convexity, elongation
- connection: neighboring elements

All the attribute values are normalized by their variances. The *connection* item reflects the relationship with other graphical objects nearby, therefore more global structural information can also be described by this set of attributes.

4.4 Interactive Recognition System

Interactive recognition has been proposed to ease the efforts of manual operation for processing results of automated recognition[62]. Generally, there are two types of manual operations: (1) correction of recognition errors, and (2) completion of processing for rejected graphical elements. Here, the first type of operations contain visually scanning through all processed results to find out all types of errors, and correcting the identified errors; the second type of operations contain identifying the class to which the unprocessed element belongs and processing the element according to its class. By statistics from practical systems, both processes take more time than automated recognition process, with the ratio varying with the quality of drawings.

To improve the reliability and efficiency of recognition system, the proposed system uses strictly conditioned rules in automated modules for patterns with little noise and distortion, so that automated recognition will generate more reliable results. At the same time, for elements rejected by automated modules, use more flexible rules to generate proposals of possible recognition results, in the order of their plausibilities. Here, flexible rules can be

formed in the following ways:

1. relaxing the threshold or range condition by widening the value ranges

2. relaxing the condition part by reducing the number of conjunctive conditions

The idea of using flexible model for matching objects has been proposed and applied to processing imageries such as cephalograms and angiograms [42],[43]. The functionality of the flexible models proposed is to pick up the candidates that are rejected by models for ideal situations. Other sets of models that continue to process the newly obtained candidates have to be designed. They are application dependent and inflexible in the sense that different models have to be designed for different type of noises. Therefore these approaches are only effective for specific targets.

In the interactive system proposed in this section, the proposals are presented to operator for confirmation. If the proposal with the highest plausibility is correct, the operator only needs to select and press a *yes* button on the screen. Otherwise, the operator has to move the mouse cursor to other proposals and press the button for confirmation. Manual correction is necessary only when all the proposals fail. In this way, the reliability of both recognition and interactive process, and the efficiency of interactive process can be greatly improved. By experiments with actual map drawings, the correct rate of recognition proposals is about 70%. Since interactive operation plays a very important role in this system, such type of system is called *interactive recognition system* hereafter.

The existing interactive recognition system makes use of production rules with fixed thresholds for generation of proposals. The shortcomings of this method are the application

dependency and ignorance of operator's response. As a result, the order of the proposals is always the same, no matter what data have been processed previously, even when very similar ones have been processed and the proposals were wrong. Besides, the thresholds used for generating proposals also need trial and error efforts and are drawing dependent.

4.5 Learning Based on Example Space

To solve the problems of the existing system stated in Section 4.4, a new method for generation of proposals of recognition results is proposed in this section. The new method makes use of the processed data to predict the possible recognition results, with higher and higher correct rate when more and more data are processed. At the same time, since there is no need for repeated processing of similar elements, a planning mechanism is also proposed for determining the order of data to be processed. When the target data has similar patterns, the correct rate of proposals can reach constant 100% after enough data have been processed, which means that the system does not have to process the rest data interactively at this point, and can switch to automatic process using the obtained examples. In this section, the new method of proposal generation and the planning mechanism is discussed in detail.

4.5.1 A New Method of Proposal Generation

In the proposed system, the already processed data are used for generation of proposals, instead of fixed thresholds. Generally, the distribution of feature values is unknown. Besides, since the rejected elements usually have different types of noises/distortions, they are generally not linear distinguishable, which means distinguishable by simple explicit

thresholds. Decision tree might be a good solution if all the data's recognition results are known[34],[35]. Even so, the processing time becomes an extra burden, when the process is an incremental one and the number of data is large.

Here, the recognition result of an unknown element is predicted by its similarity with processed elements. The similarity is based on the Euclidean distance between elements. In the prototype system, all the graphical elements are represented by a set of continuous-valued features. Previously processed elements are taken as example data and sorted in the example space $\{E^{(i)}\}$, where i is the class number, $E^{(i)}$ contains all the elements $\{e_{i1}, ..., e_{im}\}$ belonging to class i. Assume g_k is the kth feature of unknown element e, $\{f_{jk}^{(i)}\}$ the kth feature of jth example in class i. When using nearest neighbor method, the distance between unknown element e and class i can be calculated as follows:

$$Distance(e, E^{(i)}) = Min_j \left\{ \sqrt{\sum_k (g_k - f_{jk}^{(i)})^2} \right\} \tag{4.1}$$

The above equation indicates that the class of the unknown element is taken as that of the one which has the smallest Euclidean distance from it. Here, since root operation does not affect the final result, it is abbreviated in actual computation to reduce processing time.

It has been shown that when the distance between elements of the same class is smaller than that of different class, nearest neighbor will produce the best classification result[71]. Yet in general this condition does not hold. Therefore, in the actual system, in order to predict the class of an unknown element, k-nearest neighbor method is adopted in 3 steps: (1)find 6 elements that are closer to the one under consideration than the rest, (2)count the

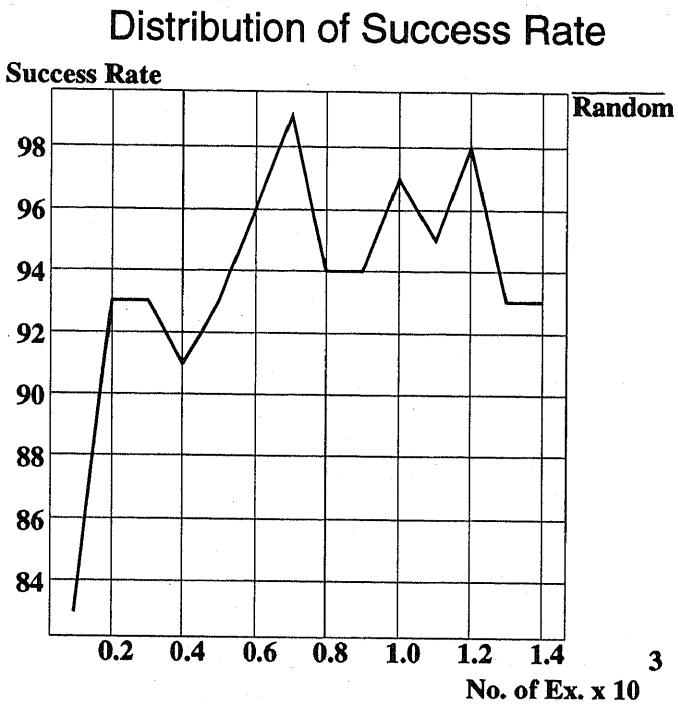


Figure 4.4: Learning curve by using example space

number of elements belonging to each class, weighted by the reverse number of distance, (3)assign the class i which has the largest weighted number of elements as that of the unknown element. Here, the number of 6 neighbors is determined through experiments.

At the beginning when there are no any elements processed, the example space is empty, therefore the system can not generate any proposal. After the first one is processed, the system can start to predict the recognition results according to the above-mentioned method. Intuitively, the more elements processed, the more possible cases will be covered by the obtained example space, and consequently the correct rate will become higher and higher. This has been proven to be true by the experiments. Therefore, the new method of proposal generation realizes a general purpose learning from examples.

Fig.4.4 shows the learning curves of the interactive recognition process. The axis of

abscissas is the number of elements processed, the axis of ordinates is the success rate of proposals. Here, success rate is defined as the number of correct proposals during processing of every 100 elements. The curve in solid line shows the success rate. The correct rate gets higher and higher with the progress of interactive recognition, The average success rate is 90%, which is 20% better than when using fixed parameters. The problem with this method is that the correct rate always fluctuates between 94% and 98% even when most of the graphical objects have been processed. This can be solved by the planning algorithm stated in next sub-section.

4.5.2 A Planning Mechanism for Stable Increase of Correct Rate

By the new method of proposal generation stated in Section 4.5.1, the proposals will be adjusted according to the previously processed elements, and the correct rate will become higher and higher with the progress of interactive processing. But one possibility is that the increase of correct rate might be unstable. Usually some of the elements are similar to each other. This implys that if dissimilar elements are processed in later stage, the obtained example space will not be good enough to cover the dissimilar examples until some of them are processed and stored in the example space. As a result, the whole set of elements have to be processed to ensure perfect recognition result. To realize a steady increase in correct rate of proposals, a planning mechanism is proposed for determining the order in which elements are processed. Assume the example space is $\{E^{(i)}\}$, the unknown elements are $\{E'^{(i)}\}$, the proposed planning mechanism chooses the next element for processing by the following evaluation:

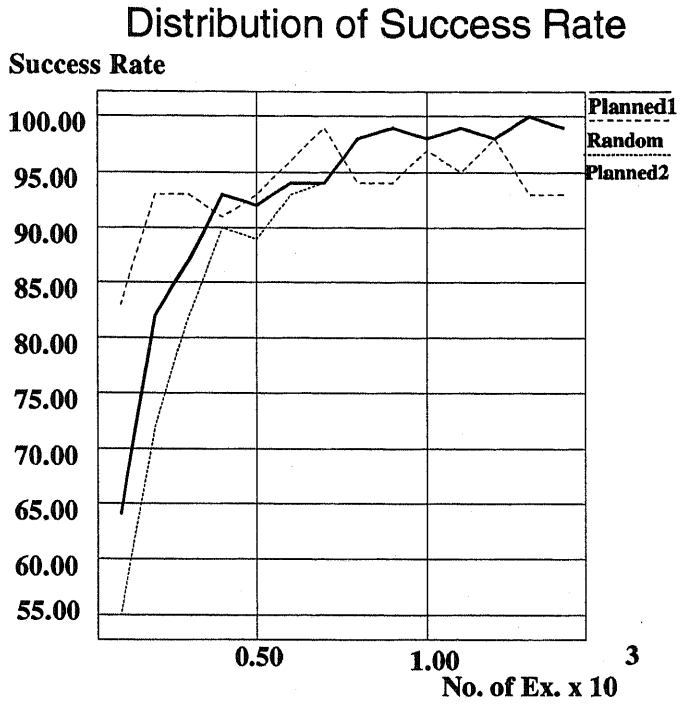


Figure 4.5: Learning curves during interactive recognition

$$Max_j\{Min_k\{Distance(e_j, e_k)\}\} \qquad (e_j \in \{E'^{(i)}\}, e_k \in \{E^{(i)}\})$$

Here, $Distance(e_j, e_k)$ is similar to equation(4.1), but has no minimum evaluation. To save processing time, all the distances can be computed in advance and stored in a table.

When the distance between two elements reflects their similarity, the above evaluation implys that the element most different from the processed ones will be chosen as the next processing target.

Fig.4.5 shows the learning curves of the interactive recognition process, when the planning algorithm is introduced. The axis of abscissas is the number of elements processed, the axis of ordinates is the success rate of proposals. Here, success rate is defined as the number

of correct proposals during processing of every 100 elements. The average success rate is about 90%. The curve in dotted line shows the success rate without planning. The curve in solid line and labeled as "plan 1" shows the result of planning. The planned result shows a steady increase of success rate, when compared to the fluctuating one by random selection labeled as "random". After about half of the elements are processed, the success rate is always above 98% by planning. Whereas without planning, the rate is always fluctuating between 94% and 98% as shown in dotted curve. The curve in dotted line and labeled as "plan 2" is the result by nearest neighbor, which has lower success rate at the beginning stage than that of k-nearest neighbor method. Therefore, the rest of experiments will adopt k-nearest neighbor method for similarity evaluation.

Fig.4.6 shows another map drawing, which has many hatching lines and comparatively less elements of other classes, and is digitized in different environment from the drawing shown in Fig.3.5. Fig.4.7 shows the corresponding learning curve during interactive recognition process. It shows that only 400 (out of 2500) elements need to be processed before the success rate becomes a constant 100%, which means that only less than one sixth of elements are required to be processed before the system can switch to automated recognition with the obtained example space. Whereas for random selection of processing targets, the success rate always fluctuates between 98% and 100% even when most of the elements are processed.

Another experiment is for isolated symbols in Fig.3.5. When a symbol does not touch any other symbols or lines, it is digitized as contour polygons. There are 10 classes of such kind of symbols and all the isolated polygons are represented by 12 shape features. Every class

has its own distortions and variations in shape. Therefore, construction of explicit rules for classification of these symbols is also difficult. Fig.4.8 shows the learning curves for the symbols in Fig.3.5. The total number of symbols is 1940, but only 400 symbols need to be processed to obtain the classification result of all the symbols. Since the isolated symbols is not affected by the overlapping with lines as the glue vector is, it is possible to induce a set of production rules by applying ID3 like algorithms to the obtained example space, even though the production rules obtained are more complex than those for ideal cases. The actual implementation of rule acquisition for this type of symbols will be discussed in Chapter 5. In this case, the interactive recognition system only serves as a tool for creating teacher data.

4.5.3 Optimization of Example Space

When the number of data processed becomes larger and larger, the size of example space will become larger and larger too, if all the processed elements are stored in the example space. A large example space will in turn cause the response time to slow down. On the other hand, the patterns of target elements usually have similarity among some of them. Here similar elements means that the Euclidean distance between them are small. Since the proposals are generated by judging the distances, similar elements will generate similar proposals because the distance among them will be close. In other words, it is not necessary to store all the processed elements into the example space.

If the border between different classes is available, the example space consisting of all the elements on the border line can ensure perfect proposal generation. Because by definition, border elements are always those that have the closest distance to elements belonging to

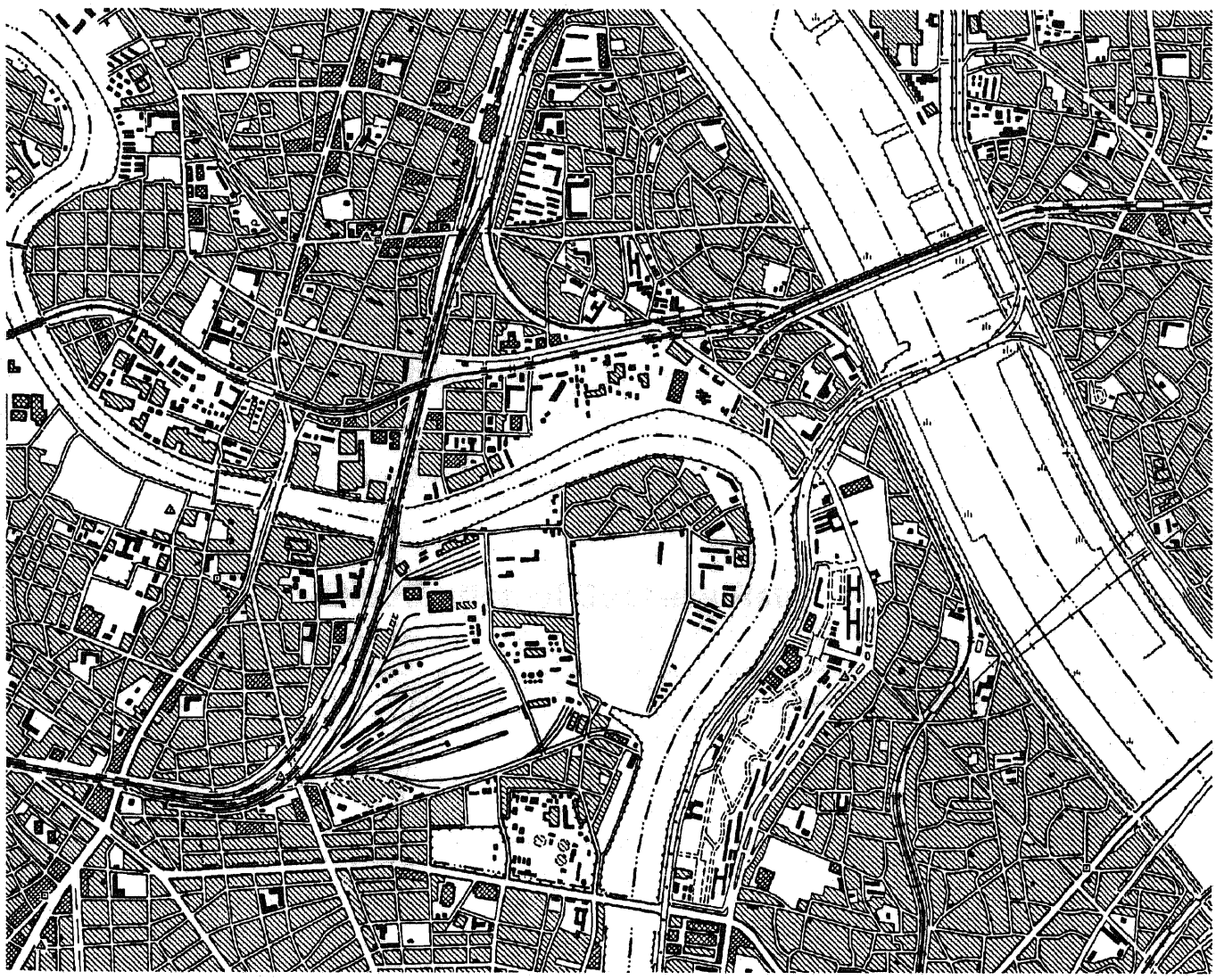


Figure 4.6: Another map for experiments

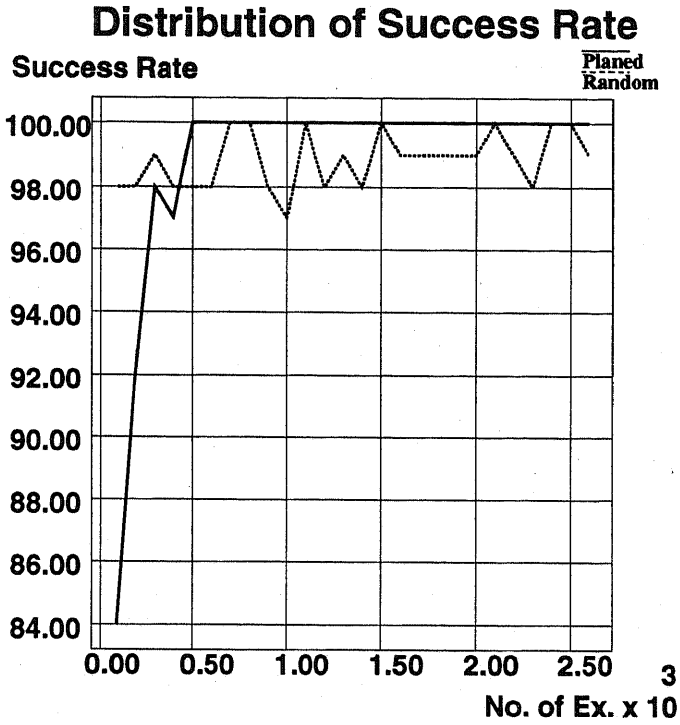


Figure 4.7: Learning curves for Fig.4.6

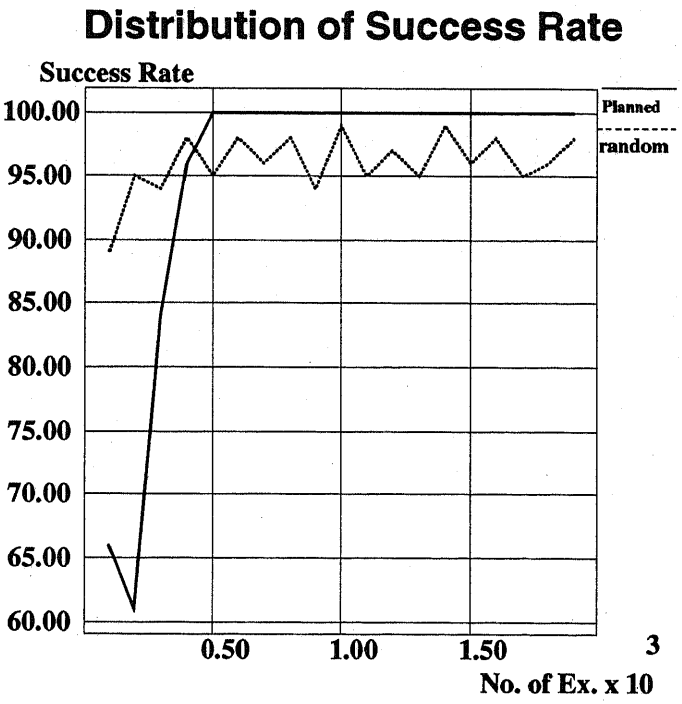


Figure 4.8: Learning curves for symbols in Figure 3.5

other classes.

When the example space is described by a 2 dimensional space, i.e. described by two attributes, the border between classes can be obtained easily by finding the corresponding Voronoi diagram. In this case, it is unnecessary to store the example data that only has Voronoi neighbor of the same class. When the example space is described by 3 dimensional space and above, i.e. described with more than 3 attributes, the computation time for obtaining Voronoi or the equivalent Delauney net will be far from practical. For example, the construction of Voronoi for a set of 500 3 dimensional data takes about 1500 seconds by an efficient algorithm[50]. Here, a more practical approach is taken by using a distance threshold D, which can be determined by the statistics of distances between all elements. The criterion for choosing the elements to be stored in the example space thus becomes: (a)if the distance between the element being processed to the existing example space is greater than D, or (b) if the proposed recognition result is wrong.

Fig.4.9 shows the result of applying the above mentioned criterion to interactive recognition process. The target elements are the glue vectors in Fig.3.5. The axis of abscissas is the number of graphical elements processed, ordinates the success rate (shown in dotted line) and the number of examples added to the example space during interactively processing every 100 elements (shown in solid line). The number of new examples added to the example space gradually redues after about 600 elements are processed. After 800 elements are processed, there are very few elements added. All through the process, the success rate of proposals is the same as when storing all the processed elements into the example space.

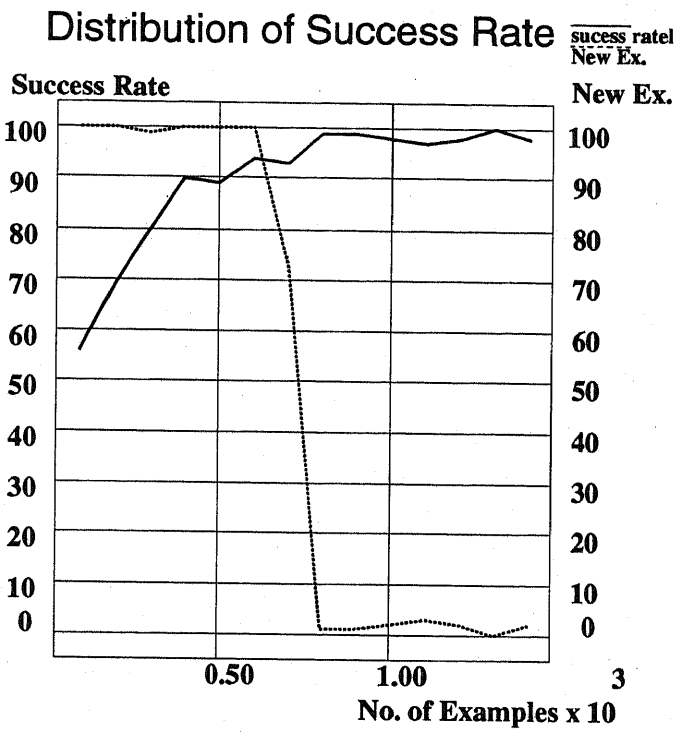


Figure 4.9: Optimization of example space

4.5.4 Application of Example Space

The obtained example space can be used in many ways to improve the processing efficiency for other drawings of the same type. The following possibilities have been considered and verified through experiments:

1. divide the example space into sub-spaces

2. if possible, generate explicit rules from the example space

3. use the example space to generate proposals for drawings of the same type

4. use the example space to train a neural network

The above mentioned possible applications of the obtained example spaces will be discussed in detail in this section.

After processing one drawing, the obtained example space can be used to process drawings of the same type and vectorized under the same conditions, such as scanning resolution, vectorization algorithm or parameters. When the distribution of the attributes used for describing the elements is not over inter-overlapping with each other, the example space can be divided into subspaces. Typically the division is n-dimensional hyper ball, which can be obtained by the following procedures when the example space is E:

1. take an example e from E
2. sort the distance from e to other examples in E in ascending order, put the result in the list L
3. find the fist example e_o that belongs to other class than e does and has the highest order in L.
4. count the number N of examples before e_o in L
5. repeat step 1 to find the e_m that has the maximum N.
6. get the distance R_m from e_m to the corresponding e_o, form a hyper ball centered at e_m and with radius R_m.
7. delete all the examples contained in the newly obtained hyper ball from E
8. repeat step 1 until E is empty

When an unknow graphical element e_x is encountered, the distance from e to the centers of the hyper balls are calculated. If e is contained in one of the balls, the corresponding class will be assigned as e's class. Otherwise, the class of the closest ball will become that of e.

For glue vectors in the city map drawing shown in Fig.3.5, a total of 151 hyper balls are generated by the above procedures. Therefore, a maximum 151×21 multiplications are necessary for proposal generation. When the hyper balls are sorted by the number of examples contained in each ball in ascending order, and B out of the total balls are used, the **cover rate** of B hyper balls can be defined as N_1/N. Here N_1 is the total number of examples contained in the B hyper balls, N is the number of total examples. Fig.4.10 shows the relation between cover rate with the number of hyper balls. The axis of abscissas is the number of hyper balls sorted by the number of examples contained in the balls in ascending order, the axis of ordinates is the coverage rate. This evaluation is for the situation in which some of the examples are exceptional ones so that referring to them might cause wrong conclusion. When this happens, the cover rate can reflect the certainty factor of the example space. From Fig.4.10, 100 hyper balls out of 151 contain 95% of the examples. Therefore, when 100 hyper balls are used for generation of recognition proposals, the certainty degree is 95%.

When the number of hyper balls obtained from the above procedures is too large, the whole example space can be used directly to generate recognition proposals. The correct rate of proposals for the followings drawings will become heigher than in the initial stage. In fact, the example space obtained during processing the drawing shown in Fig.3.5 has

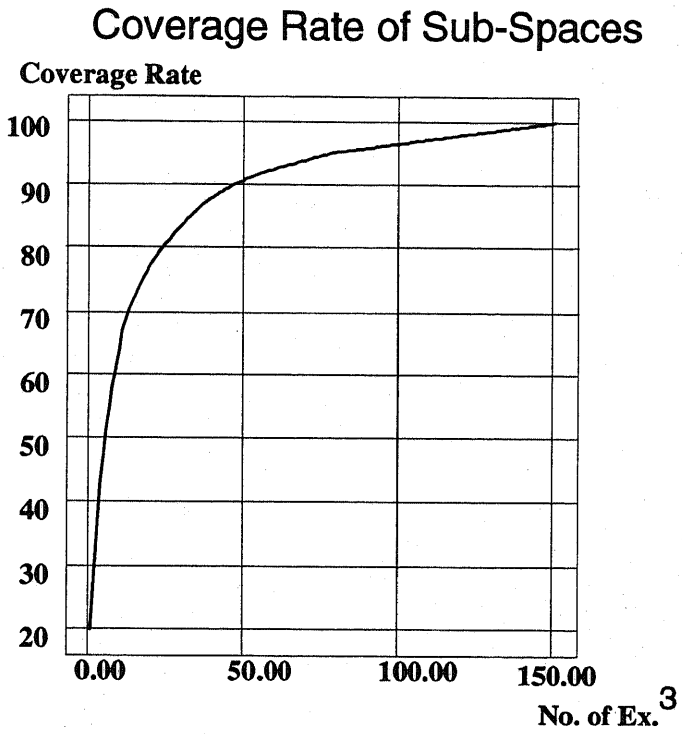


Figure 4.10: Coverage of example sub-spaces

been used to generate proposals for the drawing shown in Fig.fig-data-tokyo, which is scanned and vectorized under slightly different conditions. But still the average success rate is about 88%, which is higher than the 70% when using fixed parameters. To improve the efficiency of searching for nearest neighbor, the example data can be inserted into more efficient spatial data management structure. In this research, an efficient N-dimensional data structure GBD[67],[27] tree is used for this purpose. An experiment using GBD tree for nearest neighbor searching is conducted with the actual glue vectors in Fig.3.5 which are described in 21 dimensional parameter space. The relationship of searching time with number of example data is shown in Fig.4.11. The axis of abscissas is the logarithm of the number of examples in the example space, the axis of ordinates is searching time for nearest neighbor. Experiments are conducted with simulated data that have up to 20% changes in distance from example data, and the search time is the average of 1000 simulated data. The experimental result shows that the searching time has linear relationship with the logarithm of the number of example data. Therefore, it is practical enough to use the whole example space for proposal generation when managed by GBD tree data structure.

4.6 Application of Planning Algorithm to Training of NN

Neural networks(NN) have powerful recognition ability even for irregularly distributed patterns and have been successfully applied in recognizing symbols in drawings[26][25]. After having processed certain amount of example data in the interactive recognition processing, the obtained example space can be used to train a neural network. The trained result can then be used for generating proposals of recognition result. One of the problems with

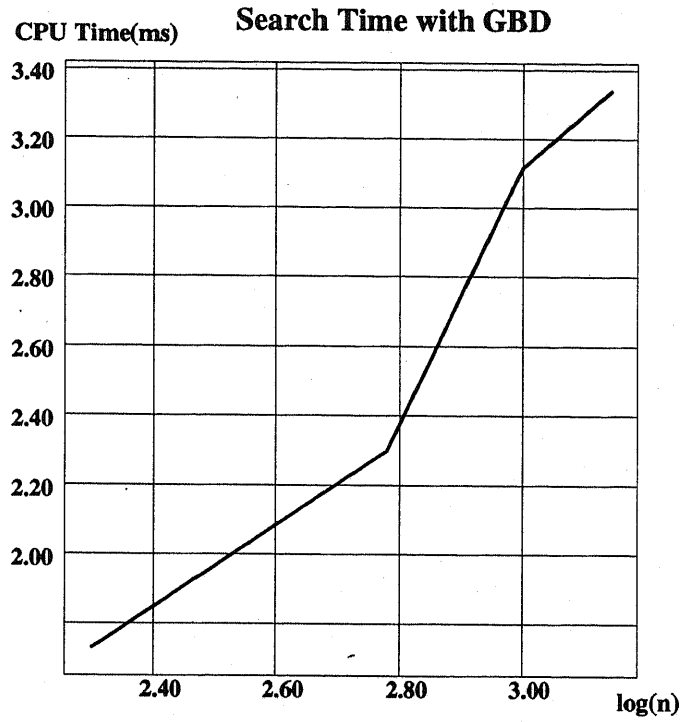


Figure 4.11: Searching time on example space with GBD tree

neural network is its time consuming process for determining the optimal structure of the networks. According to the general heuristics about determining the structure of a neural network, when there is one hidden layer, "the size of the first hidden layer is generally recommended as between one-half to three times the size of the input layer". If a second hidden layer is present, "you may have between three and ten times the number of output neurons. ··· The best way to determine optimal size is by trial and error"[70]. In the case of glue vectors stated above, the size of the input layer is 21, therefore the size of the first hidden layer can be from 10 to 63. When the size of hidden layer is 20 and number of teacher data is 1500, the typical time is 30 minutes for 3000 cycles on a Sun Sparc workstation. Therefore, to determine the best network structure, many trial and errors have to

be repeated.

In this section the result of planning algorithm stated in Section 4.5.2 is applied to teacher data. So that only objects that are different from each other are used for initial training to determine the optimal structure. As a result, the training effort can be reduced by the ratio of selected data to the total data.

To select representative teacher data, first, the teacher data set T (normalized to [0,1]) is sorted by the following steps:

1. initialize the result space R as {}.

2. find the data t that has the furthest distance from R.

3. append t to R

4. if T is not equal to {}, repeat step 2

5. end of sorting.

Next, choose objects that have smaller distance than 1 as the selected teacher data set which is then used for determining the best configuration of the neural network.

Finally use the full set of teacher data to continue training the network determined by the selected teacher data.

Fig.4.12 shows the result of neural network training with planned teacher data. A back propagation network with 21 input, 1 hidden layer of 18 neurons and 4 output is used. Out of 1482 examples, 683 are selected by the planning algorithm stated in Section 4.5.2 as teacher data. The solid curve is the change of recognition rate after every training cylce

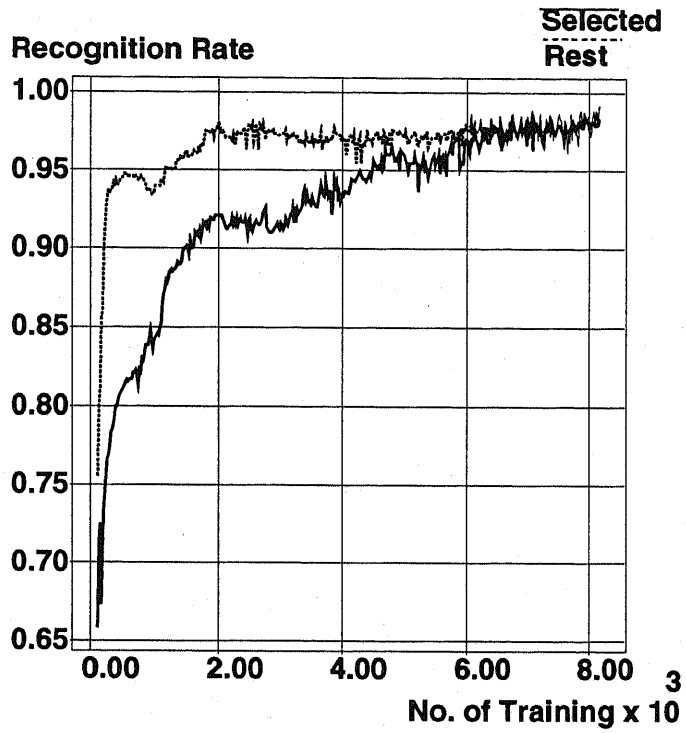


Figure 4.12: Training curve of NN with planned teacher data

during training with selected teacher data. The dotted line is the recognition rate of the rest of data using the weights trained by the selected teacher data. Since the recognition rate for the rest of data is always higher than or equal to that of teacher data, the selected teacher data is proved to be good enough to represent the whole example space and effective for training neural network.

4.7 Other Considerations

Since the proposed algorithm is based on comparison of Euclidean distance between examples, there will be many redundant distance computation if no optimization is done. When the data distribution of certain class consists of multiple condensed subspaces, the way of searching for most different element can be conducted in more effective ways. Besides,

when the number of data is too large, the computational time will become longer. To solve the above problems, the follow approaches are adopted and verified through experiments:

1. store the distance in a distance table so that the system only needs to calculate all the distances once. Then sort the distance table so that only comparison operation is necessary.

2. use density of element distribution for initial candidate selection

3. if the number of data is too huge, use k-d tree for initial candidate selection

The first approach is self explanatory. The second is expected to be effective when the target data concentrate in multiple subspaces. In this case, the most representative elements will be those at the center of the subspaces. These centers can be found by calculating the density of elements around every example data and select those with the highest density. With the sorted result by approach 1, the density can be calculated at high speed by counting the number of elements until the distance from the element under consideration is above certain threshold (1 in the experiments).

When the number of data is comparatively large, the computation time can be longer at the beginning stage of the processing because a large table will be necessary. In this case, the example space can be managed by *k-d* tree and selection of most dissimilar objects can be conducted on it. A *k-d* tree is constructed by recursively dividing the number of data in every dimension into two groups with equal number of data. The nodes in a *k-d* tree always have two children which have the same number of data when summing up all their

children underneath. When the *k-d* tree is an n-package one, the center of the elements in the leaf can be taken as dissimilar data at the initial stage of interactive processing.

Both approach 2 and 3 have been tested with the sample data used in previous sections. The typical time for generating *kd*-tree with 2000 data is less than a second on Sun Sparc workstation. Very similar results, i.e. success rate distribution during interactive recognition, have been obtained as when using the planning algorithm only.

4.8 Summary

In this chapter, a new interactive recognition system for drawings have been proposed. An interactive recognition system processes patterns with little noise/distortion by automated modules; the rest with interactive module which reduces the burden of operator by generating the proposals of possible recognition results and only leaves the final judgement to operator. The proposals for possible recognition results are formed by referring to the previously processed objects to realize a learning-from-example system. A planning algorithm have been also proposed for determining the order in which the target objects are processed. By this planning algorithm, it is possible to maintain the success rate of proposals at a steady rise and consequently reduce the potential redundant processing. The application of the obtained example space to other drawings of the same type is discussed. The planning algorithm is also applied to training of neural networks .

Experiments with practical drawing data show that the proposed approaches can obtain average success rates of above 90%, which is about 20% better than the existing approach. When the target objects have similar patterns, planning algorithm can help to reduce

the redundant processing, particularly above 80% savings for two types of data. Since the proposals are based on example space, this approach does not need target dependent trial and errors as other approaches do. The application of the planning algorithm to configuration determination of neural network training has also realized a 50% saving of effort for trial and errors with the graphical objects in the sample city map.

Currently, the neural network used in the experiments discussed in Chapter4.6 is of the most primitive structure. For better generalization ability, the approach proposed by Kayama can be adopted[65].

Chapter 5

Rule Acquisition for Understanding of Drawings

5.1 Objective

In Chapter 3 and 4, the efficient approaches of constructing automated recognization system and interactive recognization system have been discussed. The recognition system can be constructed, modified/enhanced for new specifications or new types of drawings comparatively more easily with the proposed approaches. On the other hand, the construction of new rules still needs many trial and errors. As summarized in Chapter 2, there have been researches on more powerful recognition algorithms and general purpose models. But generally their fundamental elements are still production like rules. For generation of this type of rules, there is a perpetual agony: the rule can only be effective under the limited known situations. No matter how carefully a rule is designed, there can always be the possible existence of negative situations. Nevertheless, one has to start from somewhere with an initial set of rules which are generated from known possible situation by trial and errors, and might need modification from time to time whenever new and unexpected situation occurs. Therefore, to facilitate easier construction and maintenance of recognition

rules, a new way of recognition construction is proposed, by making use of the technology of machine learning to acquire recognition rules more efficiently. In the following sections, a brief review of machine learning technology and some of its problems are first given. Next, the expansion of representation of graphical elements is proposed to improve the ability of structural information description for conceptual learning. Finally the application of machine learning techniques to acquisition of recognition rules for drawings and the improvement of the existing techniques are discussed [75],[77],[79],[83].

5.2 Machine Learning and Its Applications

The purpose of machine learning is to derive abstract concepts or rules from existing resources such as examples, facts and so on, so that the result can be utilized in turn to analyze the original resources. Depending on the inference method, learning can be roughly divided into direct implanting of knowledge, learning by deduction, learning by analogy, learning by induction and so on. The approach proposed in Chapter 4 is an example of direct implanting of knowledge. Learning by deduction is widely used for abstract concept sets, that is to retrieve new concepts from existing concepts. Learning by analogy retrieves new concept from existing ones that are generated for targets of different category or class. Learning by induction retrieves new concepts from given observation results. One of the most important area in learning by induction is learning from examples (also called empirical learning), where a set of concepts and corresponding examples are given and the learning mechanism induces from them their connotative definitions, usually in the form of classification rules. Compared to other approaches, learning by induction is more straight

forward and easy to realize. Therefore, as summarized in Chapter 2, there are numerous researches utilizing this approach for rule acquisition in image understanding.

When the learning target can be described by abstract concepts, the process of empirical learning becomes conceptual learning and is usually realized by searching through a hypothesis space, which is a combination of the existing concepts. Since the hypothesis can be very huge, many researches have been conducted for more efficient searching and have produced many practical algorithms, which can be grouped into specific-to-general searching, general-to-specific searching and their combinations[36]. Satoh presented an example of applying such algorithms to extracting structural description rules from graphical drawings by using graphical and spatial relations[32]. This kind of approach is based on the assumption that abstract descriptions of the targets are available and mainly depends on the description ability of abstract concepts. If the complexity of the target is beyond the description ability of the abstract concepts, it will be difficult to retrieve proper new concepts from the given examples. In Section 5.3, a set of expanded descriptions for graphical elements in drawings are proposed to improve the description ability of abstract concepts.

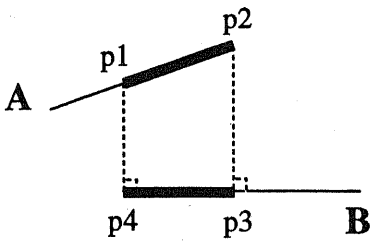
When it is difficult to describe the targets by abstract concept only, a set of attributes with nominal or numeral values will have to be used instead. This will happen frequently in the case of graphical elements in drawings. For this type of targets, a more powerful approach named TDIDT (Top-Down Induction of Decision Tree) has been proposed and has been in wide use since. The input of TDIDT is a set of classified examples described by a group of attributes; the output a decision tree that can optimally classify the given examples. One of the most successful algorithms is **ID3**, which will be discussed in detail

in Section 5.4. ID3 has been proven to be very effective when applied to the real world problems such as medical diagnosis, faulty operation diagnosis, classification of objects (glass, butterfly etc.) and so on. But so far, this type of learning problem for drawings has seldomly been discussed. Most of the researches only assume that the conceptual descriptions are available for targets that are described with numerical or nominal attributes. But in fact the process of abstracting conceptual descriptions also requires heavy efforts of trial and errors. In Section 5.4, the application of ID3 like algorithm to the problem of recognition rule acquisition for drawings will be discussed. It can be shown that this type of approach is also effective to the problem of classification of graphical elements in drawings, especially after the improvement proposed in the same section.

5.3 Expansion of Descriptions for Drawing Information

For complex objects that are composed of more than one graphical elements, it is not enough to describe them by only using descriptions such as distance between gravity centers, angles between lines or elongation axis and so on, as described in Chapter 3. In this section, the following descriptions are proposed for improvement of describing ability of understanding systems[85],[87].

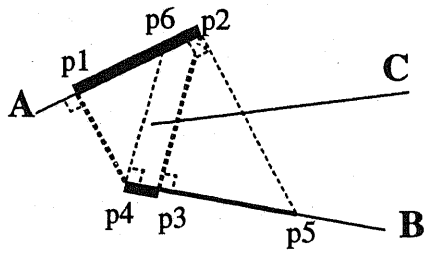
affect : As shown in Fig.5.1(a), **affect** is defined as the existence of **affected area** between two lines. Here, affected area is obtained by (a) drawing vertical lines from the end points of both line, and (b) find the intersection point (P_1 and P_3) between the vertical line and the other line. If the intersection points exist, the segments (P_1P_2 and P_3P_4)



affected area

p1p2p3p4: affecting area

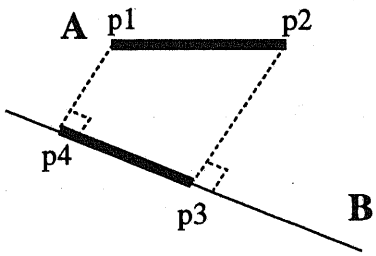
(a) Definition of "affect"

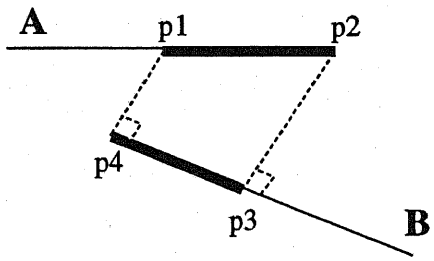


p1p2p3p4 : between area

p1p2p5p4, p3p4p6p2 : affecting area

(b) Definition of "between"



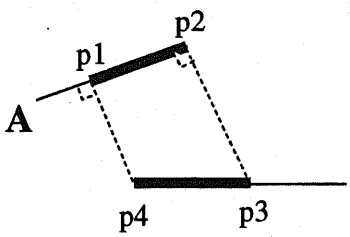


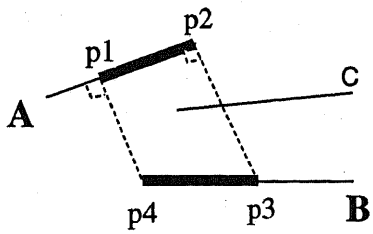
affected area

p1p2p3p4: affecting area

(c-1) above **(c-2) not above**

(c) Definition of of "above"





affected area

p1p2p3p4: affecting area

(d-1) neighbor **(d-2) not neighbor**

(d) Definition of "neighbor"

Figure 5.1: Expansion of description for complex objects

from end point to the intersection point are called affected area. Two lines are said to **affect** each other iff affected area between them exists. The polygon ($P_1P_2P_3P_4$) formed by the two end points and intersection points are called **affecting area**. The length of affected area over the length of original area is called **degree of affecting**.

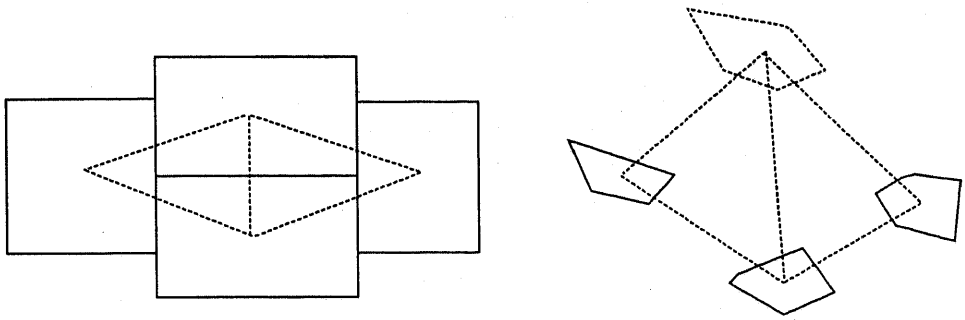
between : Line C is defined as **between** line A and B, iff A, C and B, C are affected by each other and C passes through the affecting area of A and B (Fig.5.1(b)).

symmetry : A complex object consisting of A, B, C is defined as **symmetric**, iff C is between A and B, and ($angle_{AC} \cong angle_{BC}$), and (**affected-**$area_{CA} \cong$ **affect-**$area_{CB}$).

structural triangle : When an object is composed of more than three graphical elements, **structural triangle** can be defined by connecting the gravity centers of all neighbor elements. Here when the elements are connected to each other, **neighbor** means elements that share part of the edges(Fig.5.2(a)). When the elements are separated from each other, the neighbor can be obtained by Delaunay neighbor(Fig.5.2(b)). In some cases, the shape and structural triangle can uniquely define structural information. Therefore can be useful for structural matching.

Here, the **line** in the above definition can be a physical line or the elongation axis of a polyline, polygon or complex object.

For descriptions such as **neighbor** and **above**, they can be extended as shown in Fig.5.1(c) and (d), where a neighbor of object B is defined as the object A which is within certain range of B and has no other object *between* A and B; an object A is defined as being *above* object B when A is *affected* by B and the *affected area* of A is A itself. For



(a) connected neighbor (b) dis-connected neighbor

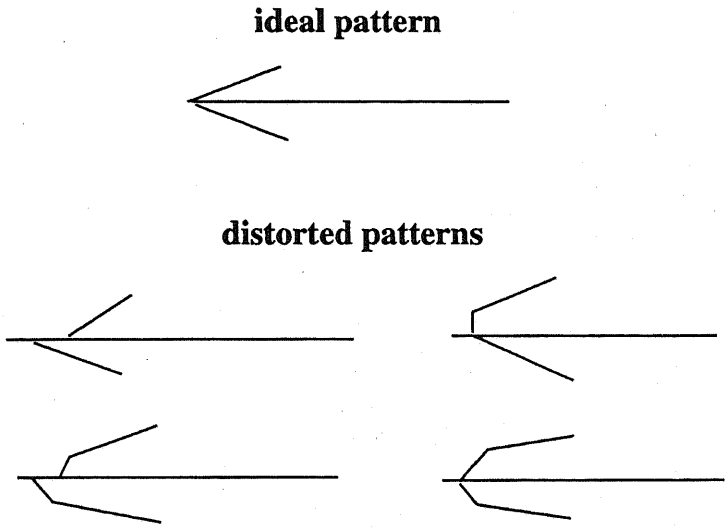
Figure 5.2: Definition of structural triangle

descriptions such as *parallel* and *vertical* lines, the certainty degree can also be defined by using the sine and cosine of the angle between the two lines respectively.

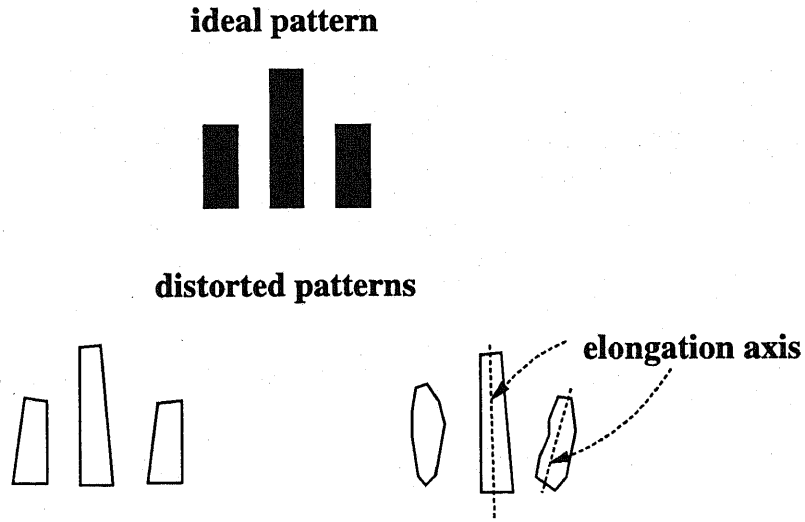
The description of affect is the fundamental spatial relationship between graphical elements in drawings. From the descriptions derived from affect relationship, it is obvious that more complex descriptions than the conventional [on top, above, below, under, to the left/right, ...] becomes possible with this fundamental description.

To show the effectiveness of the expanded descriptions, the application of some of them to two types of symbols are discussed below. The first example is the description for arrow symbols. For ideal patterns, an arrow can be defined as two short open lines with certain length, connected to one longer lines and the angle is within certain range. But for actual drawings, especially after vectorization, the ideal pattern usually becomes distored. Some of the distortions are shown in Fig.5.3(a). As a result, the thresholds for the above descriptions becomes sensitive. With the adoption of affect relation as follows, a more general and less noise/distortion sensitive rule can be constructed.

```
IF A is SHORT-OPEN-LINE          AND
```



(a) Arrows



(b) Symbol of wasted land in city map

Figure 5.3: Examples of vectorized symbols

```
    B is SHORT-OPEN-LINE          AND

    A is affected by B            AND

    angleAB < 90 degree           AND

    C is between A and B

THEN ....
```

The above rule has been applied to 7 mechanical drawings and has successfully retrieved all the 122 arrow symbols with both arrow heads correctly vectorized. Among these symbols, 72 have distortions as illustrated in Fig.5.3(a). Whereas without the expanded descriptions, at lease three rules are necessary in order to cater for the distortions.

Another example is the symbol of wasted land in city map. The ideal pattern consists of three short lines, one long and two short ones, with the longer one in between the two short ones and all parallel to each other as shown in Fig.5.3(b). When the quality of the original drawn deteriorates, the result of vectorization will be distorted as shown in the same figure. Especially conditions such as parallel become very unstable since the distortion of the contour line tends to incur a wider change in the orientation of the corresponding polygon. Therefore, the condition of parallel will have to be very loose to cover the distortions, which in turn can incur unexpected errors. In this case, the description of symmetry, between and affect can be adopted instead. Depending the actual distortions, the degree of symmetry, between and affect can be evaluated to decide the combination which is less sensitive to distortions. In this way, the rules constructed can also be more flexible and noise insensitive.

From the application result of the expanded description, we can see that when they are applied to systems such as the one in reference [32], acquisition of rules for more complex drawings can be expected.

5.4 Recognition Rule Acquisition for Drawings

In this section, the application of empirical learning algorithm to recognition rule acquisition for graphical elements in drawings will be discussed in detail.

5.4.1 Recognition Rule Acquisition through Empirical Learning

When the recognition targets are described by a set of attributes with nominal or numerical values, it is necessary to find out the appropriate combinations of value ranges of certain attributes to form rules that can classify the targets into correct classes. Since the possible combination of values is infinite with numerical attributes, the searching of appropriate combinations can not be done in the same way as in conceptual learning. When the number of available attributes and possible values are large, it is a time consuming process to find the appropriate combination. The evaluation of the result is also difficult, which means that the result might not be optimal unless exhaustive trial and error is performed. There have been many approaches proposed for this type of problems. The most successful one is to generate a decision tree by induction in a top down way.

In this study, the approach called C4 is adopted for the purpose of recognition rule acquisition. C4 is a descendant of ID3[34], which is a powerful and popular algorithm for decision tree generation in empirical learning from examples. When given a set of

pre-classified examples expressed in terms of attributes and classes, Id3 forms the decision tree by recursively selecting the attribute that minimizes the information entropy of the subset and partitioning the subset into two parts. It has been adopted and extended in many machine learning systems[35],[37],[38]. The initial algorithm can only handle nominal attributes, and extended algorithms can handle continuous valued attributes. One problem with this algorithm is that when the number of data is very large and most of data are continuous, the number of potential cut values (hereafter called **cut point**) to be evaluated increases dramatically. Fayyad has proposed a heuristic for reducing the number of cut point candidates[39]. But when the distribution is close to uniform, the reduction becomes less significant. In the case of graphical objects in drawings, most of the objects must be represented in continuous-valued attributes. And since there are different kinds of noises and distortions during conversion from image data to vector data for further analysis, the attribute values generally change continuously within certain ranges. Therefore, even with the existing improved algorithm, the improvement of efficiency is not good enough. In this section, two improved heuristics are proposed to further reduce the number of cut point candidates and ensure the optimal result at the same time. There will also be a formal proof of the proposed heuristics. The effectiveness of the proposed algorithm is verified by its application to the real-world problem, that is rule acquisition for drawing recognition.

5.4.2 Construction of Decision Tree in Empirical Learning

During decision tree construction, examples are divided into subsets according to the values of selected attribute. When continuous-valued attributes are present, they are typically discretized into subranges. The most common method of discretization is to partition the

value range into two subranges. In this case, a threshold value T is determined; and all examples that have attribute values smaller than T are partitioned to left branch of the decision tree, those greater to the right. Such a threshold T is usually called a **cut point.**

Let the subset of examples under consideration be S, K be the total number of classes, $P(C_k, S)$ be the proportion of examples in S that have class C_k, the **class entropy** of the subset S is defined as[34]:

$$Ent(S) = -\sum_{i=1}^{K} P(C_i, S) log(P(C_i, S)) \tag{5.1}$$

When the subset S is divided into S_1 and S_2 at cut point T, its *class information entropy of the partition included by* T is defined as[39]:

$$E(A, T; S) = \frac{|S_1|}{N} Ent(S_1) + \frac{|S_2|}{N} Ent(S_2) \tag{5.2}$$

The basic heuristic for decision tree construction is to find the cut point that minimizes the class information entropy defined above. It does not guarantee optimal trees, but has been proven to be very effective by its various practical applications. Obviously, in order to find the best cut point, equation 5.2 must be calculated $K \times (N-1)$ times, where N is the total number of examples in the subset under consideration. By experiments with typical drawings, dozens of classes and hundreds of examples can be quite normal. Since the learning will be done in an interactive way, fast response is required. There has been an improved algorithm for improving the efficiency of the construction of decision tree without changing the result[39].

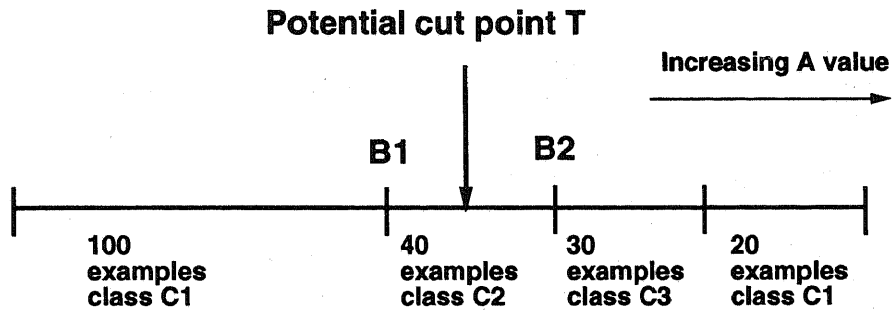


Figure 5.4: Distribution of example data when sorted by attribute A

The existing improved algorithm makes use of a heuristic (hereafter called *heuristic*1) for determining the potential cut points when attribute A_i is under consideration. It can be summarized as follows:

1. sort values of attribute A_i of all examples in ascending order

2. find the sub-ranges in which the examples belong to one single class

3. choose all the borders between the sub-ranges as cut point candidates

Fig.5.4 is an illustration taken from reference [39] which shows an example of 200 examples sorted by attribute A. The vertical arrow sign indicates a potential cut point T, short vertical bar the border between example data of different classes. It has been proven in the same reference that theoretically the class information entropy defined by equation(5.2) only takes its minimum value on the border points described above. Therefore, this heuristic can ensure the same result as when evaluating all possible cut points. Empirical evaluations show that heuristic 1 can reduce the number of cut points by a ratio of 30% to 85%.

5.4.3 An Improved Algorithm for Graphical Objects

When the number of borders defined in Section 5.4.2 is k, and the total number of examples is N, the number of cut point candidates by heuristic 1 is $k-1$, where usually $k-1 \ll N$. But when values of an attribute have uniform distribution, which means that the values are evenly distributed within a certain range, the number of borders will increase along with the degree of overlapping among classes. In the worst case, after sorting the values in ascending order, all the examples have left and right neighbors of different classes, which makes the number of cut point candidates to be $N-1$, the same as without using heuristic 1.

When attributes have uniform distribution, the number of examples is proportional to the length of the *value range* (Fig.5.5). It can be proven later that when the values have uniform distribution, the class information entropy of partitioned result always takes minimum value on the borders of a class. Simulations with presumed data show that the class information entropy is always minimum at the border of value ranges of each class. Besides, when the value ranges of each class are not too unbalanced, the overall minimum value of entropy alway falls within one of the **optimal partition regions** or is close to it. Fig.5.5 shows the definition of optimal partition regions. Fig.5.6 shows the corresponding entropy distribution. Intuitively, when cut points falls on the borders, the partition result should generate an easier to understand partition, since the cut point can obviously divide at least one class completely to one side of the partition. Generally, as long as value distribution is close to uniform distribution, the minimum value of class information entropy of partition always falls around the border of certain class.

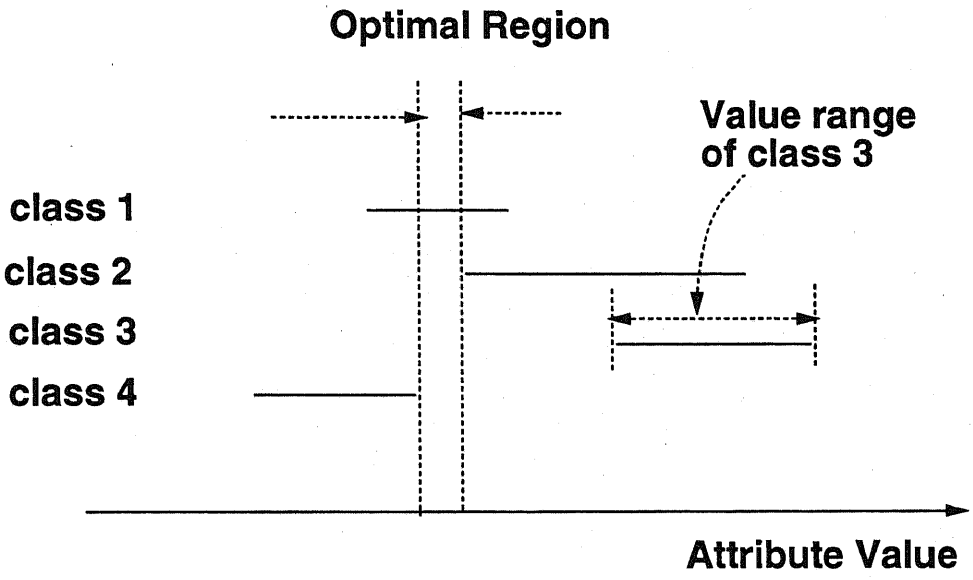


Figure 5.5: Definition of optimal partition region

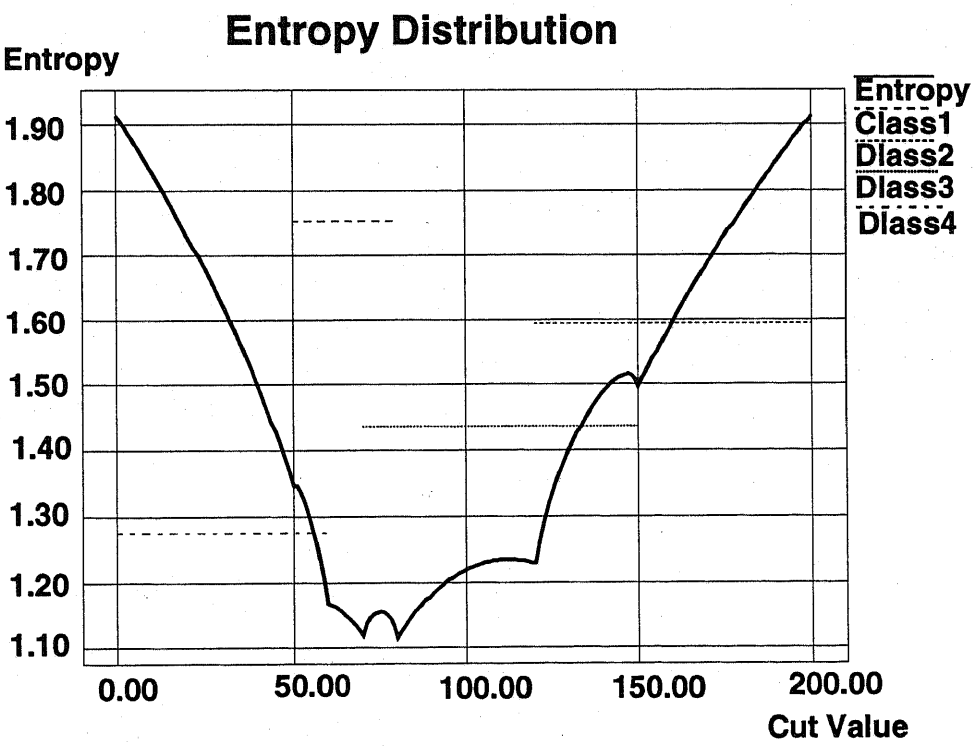


Figure 5.6: Distribution of entropy along the potential cut points

Two new heuristics

From the above results, two heuristics are proposed which examine less cut point candidates and at the same time ensure the best partition of attributes. They can be summarized as follows:

heuristic 2 when the value ranges of attribute A_i are $[min_k, max_k]$, the boundary between $v1$ and min_k, v_2 and max_k are cut point candidates. Here v_1 and v_2 satisfy the following conditions:

1. $v_1 < min_k, v_2 > max_k$
2. v_1 and v_2 belong to different class from those of min_k and max_k respectively
3. there is no other examples within $[v_1, min_k]$ and $[max_k, v_2]$

heuristic 3 when the value ranges are not over unbalanced and there exists optimal partition regions, candidates determined by heuristic 2 can be further limited to those within the optimal partition regions

Formal proof of heuristic 1

The following is a formal proof for heuristic 1 that under the assumption of uniform distribution, the class information entropy always takes minimum value on the border of value ranges. Fig.5.7, shows a general case of attribute value ranges. For convenience, let $L = \sum_{i=1}^{n} L_i + \sum_{i=1}^{m} L'_i, R = \sum_{i=1}^{p} R_i + \sum_{i=1}^{m} R'_i, N = L + R + L_0 + L_{00} + R_{00}$, $R_{01} = L_0 - L_{01}$. Then equation (5.2) becomes:

$$Ent(A, S; T)$$

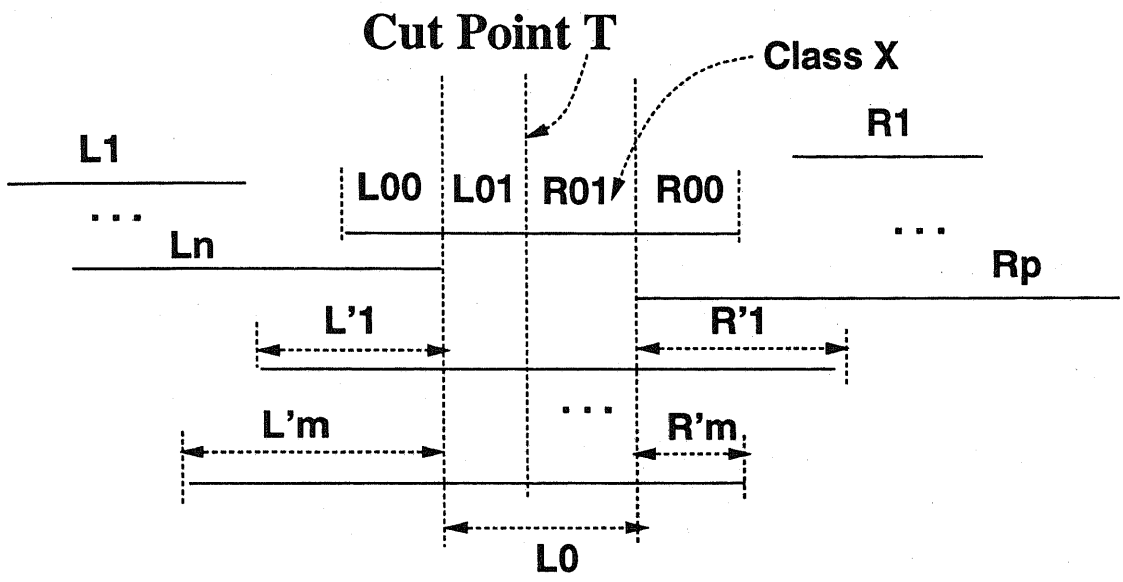


Figure 5.7: A general case of value ranges

$$
\begin{aligned}
= & -\frac{L+(1+m)L_{01}}{N}\Big(\sum_{i=1}^{n}\frac{L_i}{L+(1+m)L_{01}}log\Big(\frac{L_i}{L+(1+m)L_{01}}\Big)+ \\
& \sum_{i=1}^{m}\frac{L'_i+L_{01}}{L+(1+m)L_{01}}log\Big(\frac{L'_i+L_{01}}{L+(1+m)L_{01}}\Big)+ \\
& \frac{L_{01}}{L+(1+m)L_{01}}log\Big(\frac{L_{01}}{L+(1+m)L_{01}}\Big)\Big) \\
& -\frac{R+(1+m)R_{01}}{N}\Big(\sum_{i=1}^{p}\frac{R_i}{R+(1+m)R_{01}}log\Big(\frac{R_i}{R+(1+m)R_{01}}\Big)+ \\
& \sum_{i=1}^{m}\frac{R'_i+R_{01}}{R+(1+m)R_{01}}log\Big(\frac{R'_i+R_{01}}{R+(1+m)R_{01}}\Big)+ \\
& \frac{R_{01}}{R+(1+m)R_{01}}log\Big(\frac{R_{01}}{R+(1+m)R_{01}}\Big)\Big)
\end{aligned}
\tag{5.3}
$$

By rearranging equation (5.3)and letting $H(x) = xlog(x)$, we can get:

$$
\begin{aligned}
& Ent(A,S;T) \\
= & -\frac{L+(1+m)L_{01}+L_{00}}{N}\Big(\sum_{i=1}^{n}H\Big(\frac{L_i}{L+(1+m)L_{01}+L_{00}}\Big)+ \\
& \sum_{i=1}^{m}H\Big(\frac{L'_i+L_{01}}{L+(1+m)L_{01}+L_{00}}\Big)+H\Big(\frac{L_{01}+L_{00}}{L+(1+m)L_{01}+L_{00}}\Big)\Big)
\end{aligned}
$$

$$-\frac{R+(1+m)R_{01}+R_{00}}{N}(\sum_{i=1}^{p}H(\frac{R_i}{R+(1+m)R_{01}+R_{00}})+$$
$$\sum_{i=1}^{m}H(\frac{R'_i+R_{01}}{R+(1+m)R_{01}+R_{00}})+H(\frac{R_{01}+R_{00}}{R+(1+m)R_{01}+R_{00}})) \quad (5.4)$$

We now take the first and second derivatives of equation5.4 with respect to L_{01} and get

$$Ent(A,S;T)'$$
$$= -\frac{1}{N}(-(1+m)log(L+(1+m)L_{01}+L_{00})+\sum_{i=1}^{m}log(L'_i+L_{01})+log(L_{01})$$
$$-(1+m)log(R+(1+m)R_{01}+R_{00})+\sum_{i=1}^{m}log(R'_i+R_{01})+log(R_{01})) \quad (5.5)$$

$$Ent(A,S;T)''$$
$$= -\frac{1}{N}(\frac{-(1+m)^2}{L+(1+m)L_{01}+L_{00}}+\sum_{i=1}^{m}\frac{1}{L'_i+L_{01}}+\frac{1}{L_{01}+L_{00}}+$$
$$\frac{-(1+m)^2}{R+(1+m)R_{01}+R_{00}}+\sum_{i=1}^{m}\frac{1}{R'_i+R_{01}}+\frac{1}{R_{01}+R_{00}}) \quad (5.6)$$

Next we prove by induction that $Ent(A,S\,T)'' < 0$, when $0 < L_{01} < L_0$.

Let SUM_k be the value of $N \cdot Ent(A,S\,T)''$ when $m = k$.

When $m = 1$, the value of SUM_1 is

$$SUM_1 = \frac{4}{L+2L_{01}+L_{00}}-\frac{1}{L'_1-L_{01}}-\frac{1}{L_{01}+L_{00}}+$$
$$\frac{4}{R+2R_{01}+R_{00}}-\frac{1}{R'_i+R_{01}}-\frac{1}{R_{01}+R_{00}}$$

By rearranging the right side of the above equation, we can get:

$$SUM_1 = -\frac{L^2}{L'_{01}(L'_1+L_{01})(L'_1+2L_{01})}-\frac{R^2}{R'_{01}(R'_1+R_{01})(R'_1+2R_{01})}$$

Therefore $SUM_1 < 0$. Next assume $SUM_k < 0$,

$$SUM_{K+1}$$

$$
\begin{aligned}
= & -\frac{(1+k+1)^2}{L+L'_{k+1}+(1+k+1)L_{01}+L_{00}}+\sum_{i=1}^{k+1}\frac{1}{L'_i+L_{01}}+\frac{1}{L_{01}} \\
& -\frac{(1+k+1)^2}{R+R'_{k+1}+(1+k+1)R_{01}+R_{00}}+\sum_{i=1}^{k+1}\frac{1}{R'_i+R_{01}}+\frac{1}{R_{01}} \\
= & \frac{(2+k)^2}{L+L'_{k+1}+(k+2)L_{01}+L_{00}}-\frac{(1+k)^2}{L+(1+k)L_{01}}-\frac{1}{L'_{k+1}+L_{01}}+ \\
& \frac{(2+k)^2}{R+R'_{k+1}+(k+2)R_{01}+R_{00}}-\frac{(1+k)^2}{R+(1+k)R_{01}}-\frac{1}{R'_{k+1}+R_{01}}+SUM_k \quad (5.7)
\end{aligned}
$$

For convenience, let $a = L'_{k+1} + L_{01}, b = L + (k+1)L_{01} + L_{00}, c = R'_{k+1} + R_{01}, d = R + (k+1)R_{01} + R_{00}$. Then the right side of the above equation becomes

$$SUM_{k+1} = -\frac{(b-(k+1)a)^2}{ab(a+b)} - \frac{(d-(k+1)c)^2}{cd(c+d)} + SUM_k$$

From the assumption, $SUM_K < 0$、 and because $a > 0$、 $b > 0$、 $c > 0$、 $d > 0$, we have $SUM_{k+1} < 0$. Therefore $Ent(A, S; T)'' < 0$ for any value of L_{01} between 0 and L_0. So far we have proven that Ent(A,S;T) is convex downwards and therefore always takes its minimum value at $L_{01} = 0 or L_{01} = L_0$, which are the borders of the value range[41]. Other cases can be proven by letting L_{00} or R_{00} be 0. Hence, we have proven that $Ent(A, S; T)$ always takes minimum value at one of the borders of value range when the distributions of attribute values are uniform.

5.5 Evaluation of the Proposed Approach

In order to verify the effectiveness of the proposed approach, a prototype system has been implemented and applied to the graphical elements in city map drawings. In this section, the system configuration and experimental results will be discussed in detail. The proposed new algorithms are also compared with existing ones and the result reveals better performance.

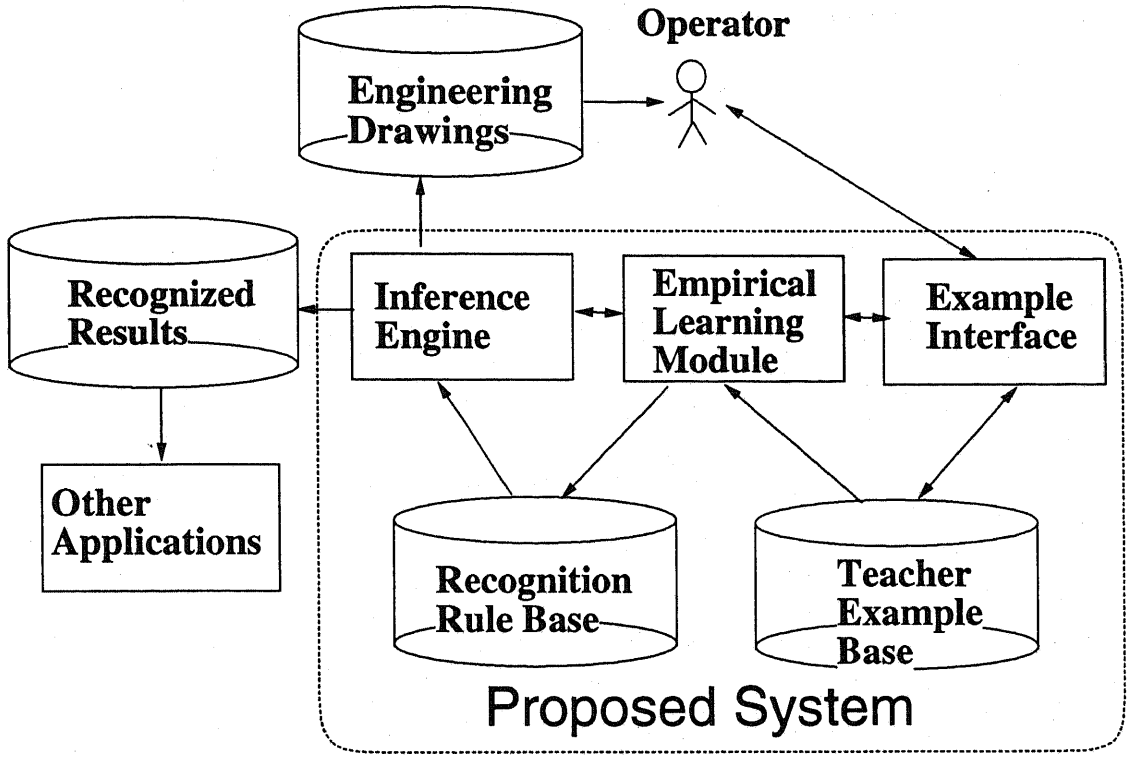


Figure 5.8: Configuration of the proposed system

5.5.1 System Configuration

The learning system is implemented on Sun Sparc workstation with C++ and Prolog language on X-window environment. The total size for decision tree generation is about 1200 lines in Prolog and 1500 in C++.

Fig.5.8 shows the configuration of the proposed system. The "Interface Engine" can be any inference systems that are driven by knowledge base. Currently, a blackboard system is being used for its simplicity and transparency. The "Empirical Learning" module constructs decision trees from given teacher data and translates the decision tree into other forms of rules, particularly in the prototype, production rules. The "User Interface" module supports operations on teacher examples and communicates with "Empirical Learning"

module. The "Empirical Learning" module shown in Fig.5.8 makes use of C4 algorithm to construct decision tree from the teacher data obtained through "User Interface" module.

At the initial stage, the operator specifys classes and corresponding examples from sample drawings to the system through *specify*, *zoom* and *scroll* functions of the user interface module. After specification of certain amount of examples, the operator can activate the empirical learning module to generate recognition rules. The generated rules are then applied to the drawings for verification of the sufficiency of teacher examples. When there is any object not correctly recognized by the generated rules, either mis-recognition or inadequate rejection, the operator can add them to the teacher examples. These procedures are repeated until all objects in the sample drawing are recognized by the generated rules.

When the obtained rules are applied to actual drawings and encounters exceptional cases that are not covered by the existing rule sets, interactive process becomes necessary to complete the recognition. In this case, the exceptional case can be added to the teacher example. The learning module can be activated again to generate new recognition rules according to the new set of examples. In this way, the modification of rule set can be easily realized.

5.5.2 Example Interface

The example interface module's main function is to supply an easy access to teacher example base. The main functions are as follows:

specification/addition of teacher examples: for specification of class name, and the corresponding objects

display of teacher examples: for confirmation of the contents of example base

deletion of inadequate examples: for correction of careless mistakes or examples that do not follow certain new drawing regulations

generation of rule: activates the learning module to generate decision trees and converts the results to rule format

verification of rule (apply): applys the generated rule to the current drawing and displays the recognition results for confirmation

file I/O: load/save of example data, load/save of rule data, load/save of drawing data

Two internal files are created by the user interface module: teacher example data(used both for rule generation and for online interactive recognition), rule data(used both for recognition and future expansion). A teacher data file consists of the followings:

Examples : class names and corresponding attribute values

Working Environment : scanning resolution, scale, drawing type and example type

The information about working environment is important when the example data is shared by similar drawings. For example, when the same type of drawing scanned with different resolution or drawn in different scales, the existing example data can still be reused by scaling the data with appropriate scale.

5.5.3 Experimental Results

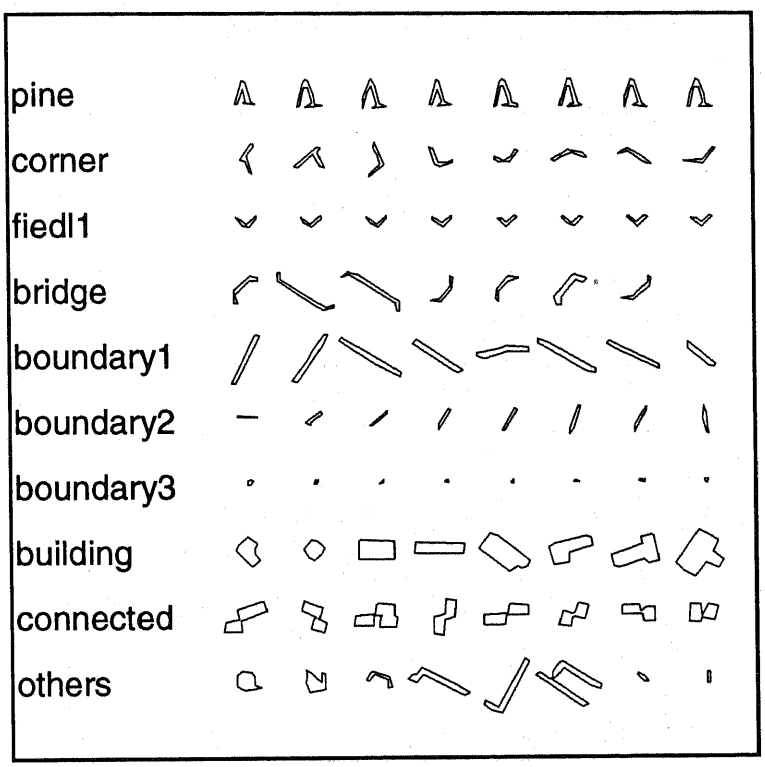


Figure 5.9: Typical Symbols with DistortionData used in experiments

To evaluate the proposed heuristics, they are applied to a real-world problem, that is rule acquisition for drawing recognition. Here map drawing is chosen as the experimental targets. The ultimate purpose of the recognition system is to convert map information drawn on the paper into machine readable abstract data. One major step of the recognition process is to classify the symbols into corresponding classes. Fig.5.9 shows some typical symbols with vectorization distortions. The symbols are designed in such way that they are visually distinguishable without much confusion. Therefore, theoretically it is possible to recognize them with a comparatively small number of concise production rules. On the other hand, the values of the attributes overlap with each other, and after vectorization, most of the symbols are distored in various ways. Besides, shapes and dimensions of some symbols such as building, bridge and corner vary within certain ranges. Therefore, tremen-

dous trial and error efforts are required for generating production rules which recognize such objects. Furthermore, whenever there is any change in vectorization conditions, symbol specification and so on, the same efforts become necessary again. As a result, more flexible ways of building drawing reconization system are expected.

By the system stated above, recognition rules can be acquired through interactive operations. An operator only needs to specify examples of every class from the display with mouse and then the system will generate recognition rules from the decision trees constructed by C4 algorithm. This approach of rule generation by empirical learning is also very effective when a new drawing is to be recognized and has some new patterns. In this case the recognition rules can be adjusted by only specifying the new pattern and letting the system rebuild the rules. Since the objects to be recognized are expressed in continuous-valued attributes and the distribution of values are close to uniform distribution, it is also a good case for verifying the effectiveness of the proposed heuristics stated in Section 5.4.3.

A map drawing is first digitized by using an image scanner. Then the line segments are detected and converted to line vectors. After vectorization, all the symbols are converted into polygons, and among many shape attributes, represented by 12 attributes as stated in Chapter 3. Here, when the target is composed of multiple elements, in this case polygons, the *area* attribute becomes the sum of all the compositional polygons. Other attributes such as convexity, rectangle likeness and so on are all derived from the compositional area attribute.

The screen layout of the implemented system is shown in Fig.5.10. Two viewing windows

are for convenience of comparing and verifying learning results. The "teacher" menu is for specification and addition of teacher examples. The popup window shows is for specification or switching of class names, where hierarchical structure is also possible. The "disp" menu is for switching view window, zooming in, scrolling etc. The "action" menu is for generation and verification of rules. Fig.5.11 shows the popup window of selection of generated rules for rule verification, where all the names of the rules are listed and can be selected with mouse cursor.

Fig.5.12 shows the teacher data of boundary symbol (left view window) and the result of recognition by the corresponding rules. Fig.5.13 shows the teacher data of building symbol (left view window) and the result of recognition by the corresponding rules. Both recognition results are not completely correct, therefore the rejected or mis-recognized ones will be added to the teacher data base later.

There are 4940 symbol candidates in the example drawings. Twelve categories are chosen for classification with two categories composed of multiple sub-parts. The learning is realized by the repetition of (a) specification of positive/negative examples, (b) generation of recognition rules and (c) verification of the generated rules until all the symbols are recognized correctly. The average number of repetition is about 8. The experiments have been carried out in three ways: (1)evaluation of heuristic 2, (2)evaluation of heuristic 3 and (3)evaluation of sensitivity of both heuristics.

When only heuristic 2 is used, the decision trees generated are completely the same as when using heuristic 1. One group of the typical experimental results are shown in Fig.5.14. Fig.5.14(a) shows the number of cut point candidates to be evaluated for best

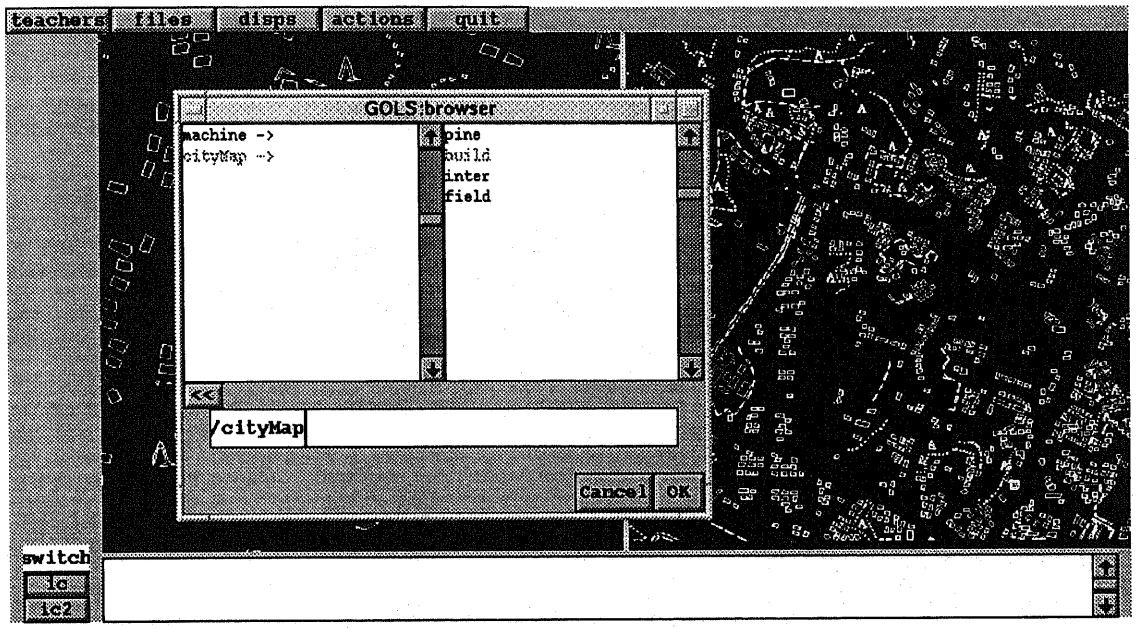


Figure 5.10: Screen layout of the proposed rule acquisition system

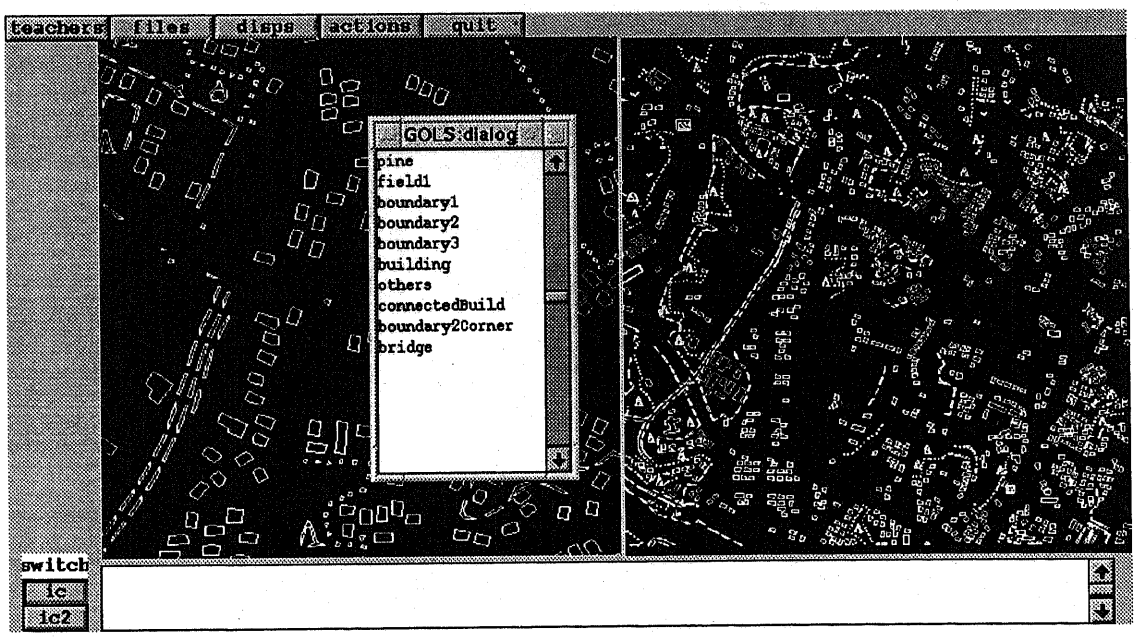


Figure 5.11: Popup window for selection of existing rules

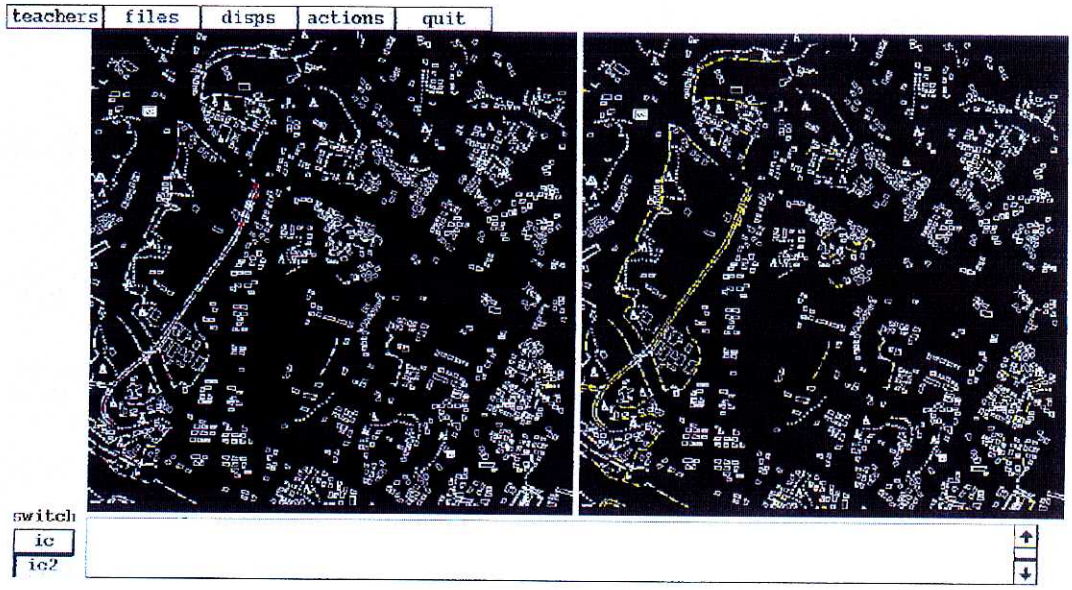


Figure 5.12: Intermediate result of learning boundary symbol

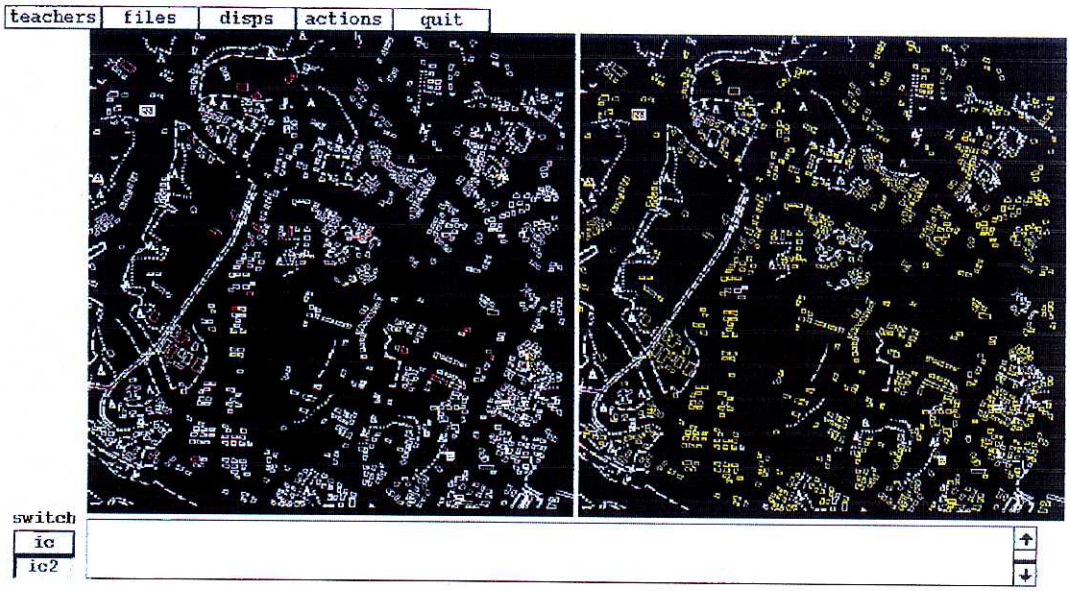
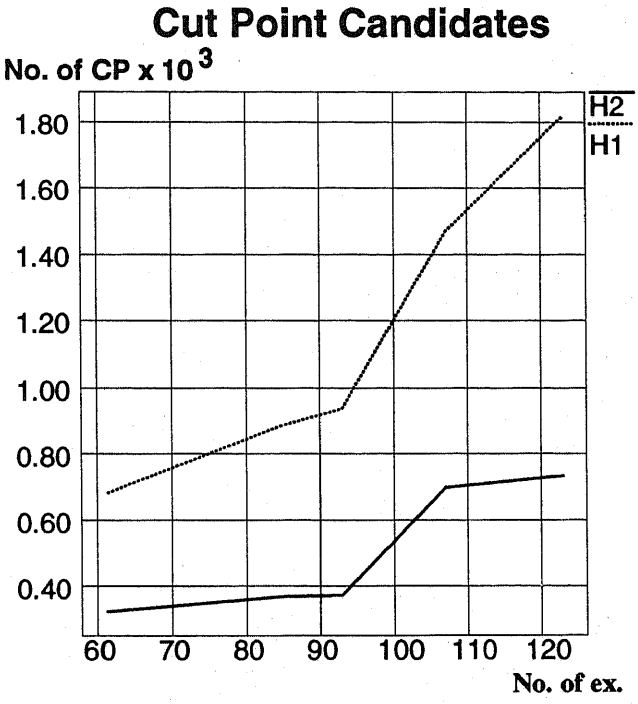
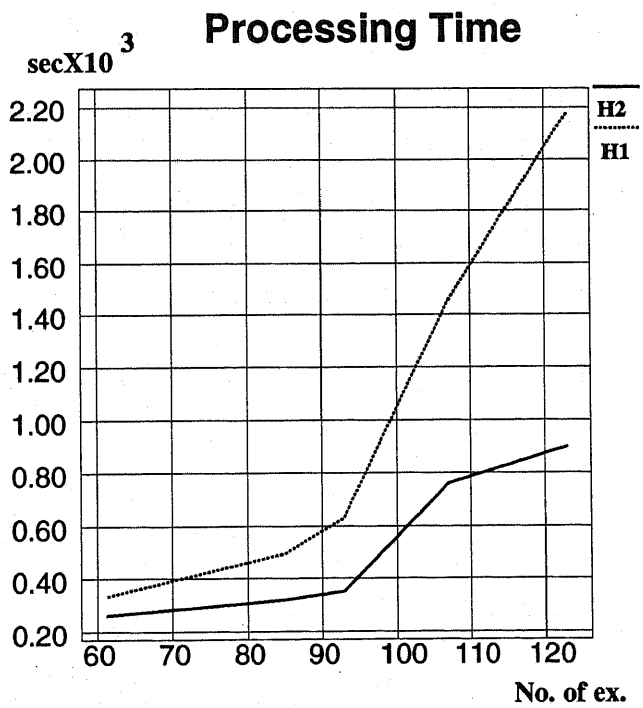


Figure 5.13: Intermediate result of learning building symbol



(a) No. of CP Candidates



(b) Processing Time

Figure 5.14: Experimental result 1

partition. The axis of abscissas is the number of examples during each repetition stage, the axis of ordinates is the number of cut point evaluated by the proposed algorithm and heuristic 1 stated in Section5.4.2. Fig.5.14(b) shows the actual processing time, including overhead. The axis of abscissas and ordinates are the number of example data and the processing time in seconds respectively. The dramatic changes are because of introduction of examples from new classes. The processing time is relatively long, because currently the Prolog has been developed in this lab to realize flexible general-purpose hypermedia development environment and currently has no compiler. It is estimated that processing time can be at least 50 times faster after compilation. Here, while processing time depends on implementation, the system has been implemented in such a way that only the criterion for cut point selection is different. From the result we can see that by heuristic 2 only, the number of cut point candidates is further reduced by 52% to 60%.

Table 5.1 shows the result of using both heuristics 2 and 3 together with other results for comparison. Five typical data sets obtained during the learning process are used. The results show that the number of cut point candidates is further reduced by 47% to 56%, making the total saving to be 74% to 82%. The decision trees generated have some differences from those by heuristic 1, but most of the trees have the same number of leaves and nodes are the same, with the exeption of the decision tree for the last set of data, which has one node and one leaf less than when using heuristic 1 or 2. For the rest cases such as data1, data4 and data5, the results are the same as before. Therefore, heuristic 3 can be said to be effective and does not affect the overall performance.

To evaluate the effect of increase in number of examples on the cut point candidates,

data set	ex. no.	class no.	number of cut points			processing time (sec.)		
			H1	H2	H2&H3	H1	H2	H2&H3
data1	61	7	680	326	173	336	204	161
data2	85	7	887	370	217	558	323	283
data3	93	7	936	374	217	630	363	320
data4	107	10	1468	736	348	1454	760	550
data5	123	10	1815	732	316	2177	900	648

Table 5.1: Experimental result 2

other sets of examples with more similar examples are tested. Here "similar example" means that the corresponding attribute values are within the same range. For comparison, five sets are chosen that have almost the same type of examples as those in experiment 2, with the total example numbers being 211, 235, 243, 257 and 273 respectively. The results are shown in Table 5.2. "type1" and "type2" mean example data sets used in experiment 1 and experiment 2 respectively. The changes in cut point candidates determined by heuristic 2 and 3 are always lower than 22% (actually only one 22% at the beginning, the rest are below 10%, whereas that by heuristic 1 is always greater than 25%. This result indicates that the proposed heuristics are less sensitive to changes in number of examples, as long as the value ranges do not change significantly. Actually most of the changes in the number of cut point candidate are caused by changes in the structure of decision trees. Since by heuristic 2, as long as the value range borders are not duplicated, the candidates should be always $2k - 2$, where k is the number of classes in the sub set of examples.

The decision trees in Table 5.2 have the same number of leaves and nodes except those for data 5. By heuristic 2 only the result has one leaf and node less than that by heuristic 1; by heuristic 2 and 3, the result has one leaf and node more but the maximum length is

data set	heuristic 1		heuristic 2		heuristic 2&3	
	type1	type2	type1	type2	type1	type2
data1	680	1175	326	403	173	181
data2	887	1273	370	404	217	215
data3	936	1290	374	370	217	229
data4	1468	1835	700	722	348	348
data5	1815	2361	732	760	316	355

Table 5.2: Experimental result 3

reduced by one, which means a more balanced tree.

Furthermore, for data set 1, the numbers of cut point evaluated by heuristic 1 and heuristic 2&3 are 680 and 161 respectively. For the data set 5, the above numbers are 1815 and 316 respectively. The change ratio in this two cases are $1815/681 = 2.67$ and $316/173 = 1.8$ respectively. Since the number of teacher ata in data 5 is about twice as many as data 1, the above analysis indicates that the increase in cut point for the proposed heuristic is much slower than heuristic 1. This is especially meaningful when the number of teacher data is much larger.

Finally, the complexity of the data set used in the experiments is evaluated by comparing the number of evaluated cut point candidates decision trees and savings of cut point with those of other systems, since the actual data sets are not available. The total number of cut points evaluated by heuristic 1 at the final stage is about 2000 to 3000. This is about the average when compared with the data sets used in reference[39](500 to 5000). The saving of cut point by heuristic 1 is about 50% to 70%. This is also about the average of those of the data sets mentioned in the same reference (33% to 84%). Therefore when judged by the degree of attribute value's overlapping, this rule acquisition problem can be considered

to be of average complexity.

5.6 Other Considerations

When the target objects can not be classified by explicit rules, i.e. the partition of teacher data according to their attribute values can never make all leaves to contain objects belonging to one single class. Even in this case, there are demands by many existing system to conduct such type of evaluation, i.e. classification by attribute values of objects. When this happens, the decision tree can be constructed in the following way:

1. restrict the depth of decision tree

   decision tree with deeper leaf usually means that either more attributes have to be used or the value distribution range of certain attribute has to be partitioned into many sub-ranges. Since the teacher data might contain some noise when no explicit decision tree can be constructed, the teacher data will become unreliable. Therefore, in this case, it is not the best solution to over partition the vale distribution range. Instead, a criterion can be set to limit the tree to certain depth. The criterion can either be an absolute value for depth or the minimum percentage of data in a subset. The classification result of such leaves is then multiple class with **certainty degree** defined below. In this way, a statistical evaluation of classification can be obtained. Other models designed target dependently can then try to identify the type of noise or reason of multiple solution.

2. assign certainty factor to the leaf in which objects of more than one class is included

When a leaf L contains objects belonging to M classes, and the number of objects belonging to class C_i is N_i, the certainty factor of L belonging to class C_i can be defined as:

$$\frac{N_i}{\sum_i N_i}$$

When similar object that does not follow the patterns contained in teacher data occurs, the rules obtained from teacher data sometimes can not cover it. In this case, the **degree of similarity** to known classes can be calculated. Assume the rule for a class C is expressed by

$[[feature_1, [min_1, max_1]], ..., [feature_n, [min_n, max_n]]]$

the degree of similarity of an unknown object O to class C can be calculated by:

$$\sqrt{\sum_i^n \left(\frac{v_i - 0.5(min_i - max_i)}{max_i - min_i} \right)^2}$$

5.7 Summary

In this chapter, a set of expanded descriptions for graphical elements have been proposed to improve the description ability of recognition systems. It can also be expected that the proposed descriptions can be applied to conceptual learning for drawings. For objects that are described with numerical objects, a rule acquisition system has been proposed. The proposed system can generate recognition rules from the given teacher data. A machine learning algorithm called C4 has been adopted for generation of decision tree from teacher data. Since the process is an interactive one, short generation time is required. To obtain faster response time, two new heuristics are proposed for determining the candidate of

optimum cut point at high speed.

The expanded descriptions have been used for constructing actual recognition rules and the results show better description ability than existing ones. For the proposed recognition rule acquisition system, experiments have been conducted with actual drawing data. The results show that recognition rules can be easily acquired by several repetition of teacher data specification, rule generation and verification. By the proposed heuristics for decision tree construction, the processing time is shortened by 74% to 82% and less sensitive to number of teacher data. The increase of cut point and processing time is proven to be at least 1.4 times slower than that of existing approach.

Chapter 6

Conclusion

Inputting paper-based drawings for database construction has been in great demand ever since all kinds of CAD/CAM system became available. Many researches has been conducted from processing of fundamental elements to understanding of drawings with specific specifications. Among many promising results, there are also many obstacles to be overcome. Aiming at general-purpose understanding system for drawings, this study has concentrated on three areas: (1) recognition knowledge representation, (2)interactive recognition and (3) recognition rule acquisition.

Knowledge Driven Understanding System

In Chapter 3 an approach has been proposed for improving the flexibility of the processing system, the recognition process is constructed by an independent knowledge base composed of structured production rule base and an blackboard model based inference engine driven by the knowledge base. By applying the implemented prototype system to several typical drawings, the following characteristics of the proposed system have been verified: (1) easy construction of recognition system, because of the usability of fundamental knowledge base and procedures (2) easy modification and enhancement of the existing system (3) possibility

of cooperative and parallel processing.

Interactive Recognition System

When the target drawings can not be fully processed by automated system, manual operation becomes necessary for correcting the wrongly recognized graphical elements and completing the processing of rejected ones. In Chapter 4 a new approach has been proposed for improving the efficiency and reliability of the manual operations. Strictly conditioned rules are used for automatically processing graphical elements with little noise to ensure high reliability. For the rest of graphical elements, use more flexiblely conditioned rules in generating proposals and let the operator do the final judgement, which requires only mouse clicking for most of the time when the correct rate of proposals is high. The proposals are generated from the processed graphical elements, which can ensure high correct rate. A planning algorithm has been also proposed to maintain the steady increase of recognition rate so that when there are repetitive patterns, there is no need to repeat the same interactive operation. The possibility of utilizing the obtained example space for drawings of the same type and the application of planning algorithm to training of neural networks have also been discussed.

Acquisition of Recognition Rule through Machine Learning

In Chapter 5, the expansion of descriptions for graphical elements in drawings is proposed in order to increase the description ability of recognition/understanding system for drawings in general. When used for conceptual learning, better description ability can also be expected. An improved machine learning technique is proposed for efficient acquisition of recognition rules. The process of writing recognition rules for graphical objects requires

many trial and errors. With the proposed system, recognition rules can be easily acquired by several repetition of teacher data specification, rule generation and verification. In this way, not only the construction process of initial recognition rules becomes easier, but the maintenance(improvement, enhancement etc.) of the rule base also becomes easier.

In conclusion, while not being the total solutions for general purpose system, experiments with representative drawings have proven the proposed approaches to be effective for improving the flexibility and portability of recognition/understanding system for drawings.

Acknowledgements

I would like to express my gratitude to professor Masao Sakauchi for his precious guidance and helpful advices during my research activities. He has also been warmly supporting me in my personal life. All his great efforts have made this thesis possible. I would also like to express my gratitude to professor Yutaka Ohsawa (Saitama University) for his help in part of my researches. He has helped to proofread some of my papers and given many useful comments. Mr. Toshiyuki Okuhashi(Marketing Intelligence Corporation) has helped in the research in Chapter 4, Chapter 3, and also in conversion of drawing data for experiments. My researches can be carried out with less efforts of building the user interface modules thanks to Mr.Takashi Satoh's great work of GOLS system. Dr. Shin'ichi Satoh (National Center for Science Informatoin System), Dr.Jun Yamane (Ricoh Co. Ltd), Mr.Yoshitomo Yaginuma, Mr.Yin-Ming Lin and Mr.Wei Wu also had many meaningful discussions with me so that the researches can be carried out smoothly. Mr.Suguru Sato, Mr.Takashi Satoh and Mr.Takeshi Sagara have also helped me a lot regarding computer system problems. I am also deeply indebted to the following members of Sakauchi's laboratory for their kind cooperations: Professor Heitou Zen (Shousen University), Dr.Junichi, Tatemura, Mr.Shin Yamada (Matsushita Co. Ltd.), Mr.Masahiro Hiwatashi (Asahi Chemical Co. Ltd.), Mr.Atsushi, Ono (Sharp Co. Ltd.), Mr.Hiroshi Mo, Mr.Naoki Nishikado, Mr.Chun-Xiao Li, Mr.Ping-Tao Wang, Mr.Junpei Sano, Mr.Shigeki Yamano, Mr.Tetsuya Katsumata, Mr.Naoki Izumi, Ms.Peilin Liu and all the secretaries. Finally I would like to thank my wife Shi Jie, my parents and inlaws for their persistent support so that I could have been concentrating on my researches.

Bibliography

[1] M. Sakauchi, Y. Ohsawa: Image Database. Shokoudo, 1987

[2] Y. Ohsawa, S. Yamakawa: Recognition and Understanding of Drawings, Shokodo, 1989

[3] R. Kasturi and L.O. Gorman: Document Image Analysis: A Bibliography, Machine Vision and Application, pp.231-243, 1992

[4] K. Tombre: Technical Drawing Recognition and Understanding: From Pixels to Semantics, Proc. of IAPR Workshop on Machine Vision Applications, pp.393-402, 1992

[5] M. Hirada: Automate Drawing Inputting System. Nikkei Computer Graphics, No.2, pp.10-25

[6] S. Hine and M. Matsusaka: Automatic Recognition System for Mechanical Drawings "ARCADIA-M", Proc. of IPS, 1986, Japan

[7] R. Kasturi, R. Fernandez, M. L. Amlani etc: Map Data Processing in Geographic Information Systems, IEEE, Computer Magazine, Vol.22, No.12, pp.10-21, 1989

[8] S. Ablameyko and O. Frantskevich: From Computer Vision to Document Recognition or Using Labeling Technique for Map Interpretation. IAPr Workshop on Machine Vision Applications, pp.255-258, 1994

[9] M. Iwasaki, M. Yamamoto et al: Development of an Automatic Input System for Mechanical Part Drawings. IEICE Technical Report Japan, pru87-24, PP.51-66. 1987

[10] T. Agui, Y. Mesaki and M. Nakajima: A Recognition for Engineering Drawings Using Double Square Moving Method. IEICE Technical Report Japan, PRU87-55, pp.77-86

[11] O. Hori, S. Shimotsuji, F. Hoshino, et al: Probabulistic Relaxation Method for Line-drawing Interpretation. Proc. of 11th IAPR international Conference on Pattern Recognition, pp.158-161

[12] C. M. Williams: An Efficient Algorithm for the Piecewise Linear Approximation of Planar Curves, CGIP, Vol.2, 1978

[13] Y. Ohsawa and M. Sakauchi: Picture Processing Using Multidimensional Data Management Structure – Vectorization of Drawings, Trans. IECE Japan, J68-D, 4, pp.845-852, 1985

[14] Y. Ohsawa and M. Sakauchi: BD–tree: A New N - dimensional Structure for Efficient Interactive graphics, Proc. of IFIP83, 1983

[15] D. Niyogi,and S. N. Srihari: A Rule–based System for Document Understanding, Proc. AAAI-86, pp.789-793, 1986

[16] Y. He and A. Kundu: 2-D shape Classification Using Hidden Markov Model, IEEE Trans. Pattern Anal. Mach Intell., Vol.13, No. 11, 1991

[17] E. E. Milios: Shape Matching Using Curvature Processes. Computer Vision, Graphics and Image Processing Vol. 47, pp.203-226, 1989

[18] N. Ansari and E. J. Delp: Partial Shape Recognition: A Landmark-Based Approach. IEEE Trans. Patt. Anal. Mach. Itell. Vol.12, No.5, pp.470-483, 1990

[19] C. S. Fahn, J. F. Wang and J. Y. Lee: A Topology-Based Component Extractor for Understanding Electronic Circuit Diagrams. Computer Vision, Graphics, and Image Processing. Vol.44, pp.119-138, 1988

[20] S. Yamagawa, Y. Oda: Interactive Inputing System for Engineering Drawings. Nikkei Computer Graphics, No.4 pp.120-130, 1987 (in Japanese)

[21] U. Landau: Estimation of a circular arc center and its radius. Comput. Vision, Graphics & Image Proces. Vol.38, pp.317-326, 1987

[22] R. L. Rosin and G. A. W. West: Segmentation of edges into lines and arcs. Image and Vision Computing, Vol.7, No.2, pp.109-114, 1989

[23] W. W. Seemuller: The extraction of Ordered Vector Drainage Networks from Elevation Data. Computer Vision, Graphics, and Image Processing Vol.47, pp.45-58, 1989

[24] K. Kise, M. Yamaoka, N. Babaguchi et al: Organizing Knowledge-Base for Structure Analysis of Document Images. Trans. of Information Theory and Algorithms, Vol.34, No.1, pp.75-87, 1993 (in Japanese)

[25] T. Cheng, J. Hkan and D. Y. Y. Yun: A Symbol Recognition System, Proc. of 2nd Conf. on Document Anal. and Recog. pp.918-921, 1992

[26] H. Arai, S. Abe and M. Nagura: Intelligent Interactive Map Recognition Using Neural Networks, Proc. of 2nd Conf. on Document Anal. and Recog. pp.922-925, 1992

[27] H. Shimodaira: Performance Texts on the GDB Tree Employing an Improved Method by Using Architectural Drawing Data. Trans. of Information Theory and Algorithms, Vol.34, No.1, pp.177-181, 1993 (in Japanese)

[28] S. Satoh, Y. Ohsawa, M. Sakauch: A Proposal of Drawing Image Understanding System Applicable to Various Target Drawings. Trans. of Information Theory and Algorithms, Vol.33, No.9, pp.1092-1102, 1992

[29] T. Hayakawa, T. Watanabe, Y. Yoshida et al.: Extraction of Topological Road-Network from an Urban Map. Trans. IEICE. Vol.74-DII,No.6,757-765, 1991

[30] Q. Luo, T. watanabe and N. Sugie: Automatic acquisition of layout knowledge for the structure recognition of table-form documents. Trans. IEICE, Vol.J76-D-II, No.3, 1993

[31] S. Satoh and M. Sakauchi: Understanding Rule Generation Supporting System for Drawing Understanding Using Interaction with User, IAAA Workshop on Machine Vision Application, pp.407-410, 1992

[32] S. Satoh, T. Satou and M. Sakauchi: One Method of Structural Description Rule Extraction Based on Graphical and Spatial Relations. Proc. of ICPR, pp.281-284, 1992

[33] M. Yachida and S. Tsuji: A Versatile Machine Vision System for Complex Industrial Parts. Trans. IEICE, Vol.J59-D, No.3, pp.149-156

[34] J. R. Quinlan: Induction of decision trees. Machine Learning 1, pp.81-106, 1986

[35] J. R. Quinlan: Decision Trees and Multi-Valued Attributes, Machine Intelligence, 11, pp.305-318, 1988

[36] S. C. Shapiro et al: Encyclopedia of Artificial Intelligence. John Wiley & Sons, 1987

[37] J. Cheng, U. M. Fayyad, K. B. Irani et al: Improved decision trees: A generalized version of ID3. Proceedings of the Fifth International Conference on Machine Learning, pp.100-108. San Mateo, CA: Morgan Kaufmann.

[38] P. Clark and T. Niblett: The CN2 induction algorithm. Machine Learning, 3, pp.261-284, 1988

[39] U. M. Fayyad: On the Handling of Continuous-Valued Attributes in Decision Tree Generation, Machine Learning 8, pp-87-102, 1992

[40] M. Maza: A Prototype Base Symbolic Concept Learning System, Proc. of the 8th International Workshop on Machine Learning, pp.41-45, 1991

[41] D. G. Luenberger: Introduction to linear and nonlinear programming. Reading, MA: Addison-Wesley, 1973.

[42] A.D. Levy-Mandel, A.N. Venetsanopoulos and J.K. Tsotsos: Knowledge-base landmarking of cephalograms. Comput. Biomed. Res. 19, 282-309, 1986

[43] P. Suetens, C. Smets, F. Van De Werf et al: Recognition of the coronary blood vessels on angiograms using hierarchical model-based iconic search. Proc. of Comput. Vision and Patt. Recog. (san Diego. Calif.) pp.576-581, 1989

[44] R. T. Chin and H. K. Wan: A One-Pass Thinning Algorithm and Its Parallel Implementation. Computer Vision, Graphics and Image Processing Vol.40, pp.30-40, 1987

[45] L. O. Gorman: $K \times K$ Thinning. Computer Vision, Graphics, and Image Processing Vol.51, 195-215, 1990

[46] S. A. Mahmoud, I. Abuhaiba and R. J. Green: Skeletonization of Arabic Characters Using Clustering Based Skeletonization Algorithm. Pattern Recognition, Vol.24, No.5, pp.453-464, 1991

[47] Y. Nagano, H. Kanechika, S. Tanaka et al: Experimental System Using an Interactive Drawing Input Method. Proc. SPIE Workshop on Visual Communications and Image Processing '91 : Visual Communication, SPIE Vol.1605, pp.614-623, 1991

[48] H. Maehara, J. Shibayama and J. Sawamoto: Vector-based Editing Method of Drawings for Facility Maintenance. Proc. of IAPR Workshop on Machine Vision Applications, pp.301-306, 1994

[49] W. Kim, Y. Hirai, T. Furukawa et al: Extraction of Road Segments by Spatial Filters. Trans. of IEICE, Vol.J76-D-II, No.3, pp.566-574, 1993

[50] H. Inagaki, K. Sugihara and No. Sugie: Numerically Robust Algorithm for Constructing 3-dimensional Voronoi Diagram. Algorithm, Vol.24, No.6, pp.122-129, 1991 (in Japanese)

[51] M. Kawashima, R. Tokunaga, Y. Hirai: Contour Map Reconstruction via Muti-module Parallel Computational Scheme, Proc. of IAPR Workshop on Machine Vision Applications, pp.87-91, 1992

[52] J. Jimenez and J. L. Navalon: Some experiments in image vectorization. IBM J. Res. Develop. Vol 26, pp.724-734, 1982

[53] J. Sklansky and V. Gonzalez: Fast polygonal approximation of digitized curves. Pattern Recognition Vol.12, pp.327-331, 1981

[54] H. Aoyama and M. Kawagoe: Piecewise Linear Approximation Method Preserving Visual Feature Points of Original Figures. Computer Models and Image Processing, Vol.53, No.5, pp.435-446, 1991

[55] F. Mokhtarian and A. Mackworth: Scale-based Description and Recognition of planar Curves and Two Dimensional Shapes. IEEE Trans. Pattern Anal. Mach. Intell. Vol.8, pp.34-43, 1986

[56] N. Ansari and E. J. Delp: On Detecting Dominant Points. Pattern Recognition. Vol.24, No.5, pp.441-451, 1991

[57] H. Yamada, K. Yamamoto, T. Saito et al: MAP: Multi-Angled Parallelism for Feature Extraction from Topographical Maps. Pattern Recognition, Vol.24, No.6, pp.479-488, 1991

[58] D. M. Wuescher and K. L. Boyer: Robust Contour Decomposition Using a Constant Curvature Criterion. IEEE Trans. Pattern Anal. Mach. Intell., Vol.13, No.1, pp.41-51, 1991

[59] T. Bjerch and T. Taxt: Syntax Analysis in Automated Digitizing of Maps. Proc. 4th International Conference on Pattern Recognition, pp.50-57, 1988

[60] D. Dori and A. Pnueli: The grammar of dimensions in machine drawings. Computer Vision, Graphics and Image Processing, Vol.42, pp.1-18, 1988

[61] J. S. Lee and Sl J. Chung: Reconstruction of 3-D Terain Data from Contour Map. Proc. MVA'94 IAPR Workshop on Machine Vision Application, pp.281-284, 1994

[62] Y. Ohsawa, Y. Takishima, M. Sakauchi: An Efficient Map Data Conversion System Using Deviced Combination of Full Automatic and Human-Assisted Recognition Phases. Trans. of IEICE, J72-D-II,4,545-554,1989.4

[63] Y. Ohsawa and M. Sakauchi: An Improvement of Vectorization Method to Facilitate Separation of Contact Symbols from Line Drawings. Trans. of IEICE, J72-D-II, 9, pp.1579-1581 (in Japanese)

[64] Y. Ohsawa and M. Sakauchi: A Tree–Type Line Data Management Structure for Efficient Interactive Graphics, Proc. of ICPR, Vol.2, 1986

[65] M. Kayama and S. Abe: Training Neural Net Classifier for Improving Generalization Capability. Trans. of IEICE, Vol.J76–D–II, No.4 pp.863-872, 1993

[66] C. Terry and W. Bischof: Learning Structural Descriptions of Pattern and Objects. Proc. of Asian Conference on Computer Vision, pp.23-25, 1993 (Japan)

[67] Y. Ohsawa, and M. Sakauchi : A New Tree Type Data Structure with Homogeneous Nodes Suitable for Very Large Spatial Database. IEEE the 6th Int. Conf. on Data Eng. 1990

[68] R. M. Haralick and D. Queeney: Understanding Engineering Drawings. Computer Graphics and Image Processing Vol.20, pp.244-258, 1982

[69] M. Onoue: Image Processing Handbook. Shokoudou, 1987

[70] V. B. Rao and H. V. Rao: C++ Neural Networks and Fuzzy Logic. MIS press, New Yok 1993

[71] T. M. Cover and P. E. Hart : Nearest Neighbor Pattern Classification. IEEE Trans. on Information Theory, Vol. IT-13, no. 1, Jan. 1967.

[72] W. Lu, Y. Ohsawa, M. Sakauchi: A Database Capture System for Mechanical Drawings Using an Efficient Multi-Dimensional Graphical Data Structure, Proc. of 9th International conference on Pattern Recognition, 1988

[73] W. Lu, T. Okuhash, Y. Ohsawa and M. Sakauchi: A Map Data Retrieval System with Learning Ability through Example Space, Trans. ITE, (submitted)

[74] W. Lu, M. Sakauchi: An Interactive Map Drawing Recognition System with Learning Ability, Proc. of IAPR Workshop on Machine Vision Applications, pp.235-238, 1994

[75] W. Lu and M. Sakauchi: A New Algorithm for Handling Continuous-Valued Attributes in Decision Tree Construction And Its Application to Drawing Understanding. The 8th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert System, 1995 (to be presented)

[76] W. Lu, T. Okuhashi and M. Sakauchi: A Proposal of Efficient Interactive Recognition System for Understanding of Map Drawings. The 3rd International Conference on Document Analysis and Recognition, 1995, (submitted)

[77] W. Lu and M. Sakauchi: A Drawing Understanding System with Rule Acquisition Ability. The 3rd International Conference on Document Analysis and Recognition, 1995, (submitted)

[78] T. Okuhashi, W. Lu, Y. Ohsawa and M. Sakauchi: Recognition of map drawings based on contour vector, Trans. IEICE D-II (in preparation)

[79] W. Lu, W. Wu, M. Sakauchi: Learing Structural Information in Mechanical Drawing. 48th IPSJ National Conf., 2-29, 1994

[80] W. Lu, T. Okuhashi, M. Sakauchi: Construction of Map Drawing Recognition System with Learning Ability through Example Space. 48th IPSJ National Conf., 2-23, 1994

[81] W. Lu, W. Wu, M. Sakauchi: An Efficient Training Method for Neural Networks Used for Map Drawing Recognition. IEICE National Conf. D-296, 1994, Autumn

[82] W. Wu, W. Lu, M. Sakauchi: A General Purpose Object-Oriented Model for Understaind of Drawings. 48th IJSP National Conf., 2-27, 1994

[83] T. Satou, W. Wu, W. Lu, M. Sakauchi: Drawing Recognition on GOLS platform. 5th Symp. of Functional Graphical Information System, 1994

[84] G. Lou, W. Lu, M. Sakauchi: Understanding of Pipe Layout Drawing. 49th IPSJ National Conf., 1994

[85] W. Wu, W. Lu, M. Sakauchi: OO-Mudams — A General Purpose Drawing Understanding System with Noise Absorption Ability. D-295, IEICE National Conf., 1994 Autumn

[86] W. Wu, W. Lu and M. Sakauchi: MTDM – An Active Model for Drawing Understanding. The 8th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert System, 1995 (to be presented)

[87] W. Wu, W. Lu and M. Sakauchi: An Object–Oriented Model MTDM for Drawing Understanding System and Its Ability of Noise Absorption. The 3rd International Conference on Document Analysis and Recognition, 1995, (submitted)